

DR



DeepRob

Lecture 3
Linear Classifiers
University of Michigan and University of Minnesota





Project 0

- Instructions and code available on the website
 - Here: deeprob.org/projects/project0/
- **Due tonight! January 12th, 11:59 PM EST**
- **Everyone granted 1 extra late token (3 total for semester)**





Project 0 Suggestions

- If you choose to develop locally
 - **PyTorch Version 1.13.0**
- Ensure you save your notebook file before uploading submission
- Close any Colab notebooks not in use to avoid usage limits





Project 1

- Instructions and code will be available on the website by tomorrow's discussion section
- Classification using K-Nearest Neighbors and Linear Models

Calendar

Week 1

Jan 5:	LEC 1	Course Introduction
	PROJECT 0 OUT	
Jan 6:	DIS 1	Intro to Python, Pytorch and Colab

Week 2

Jan 10:	LEC 2	Image Classification
Jan 12:	LEC 3	Linear Classifiers
	PROJECT 0 DUE	PROJECT 1 OUT
Jan 13:	DIS 2	Intro to PROPS Dataset

We're here!





Discussion Forum

- Ed Stem available for course discussion and questions
 - Forum is shared across UMich and UMinn students
 - Participation and use is not required
 - Opt-in using [this Google form](#)
 - **Discussion of quizzes and verbatim code must be private**



Gradescope Quizzes

- Course not published yet
- Roster will be uploaded and published by discussion section tomorrow
- Quiz links will be published at the start and end of lecture
- Time limit of 15 min once quiz is opened
- Each available to take from 3:00pm–6:00pm on quiz days
- Covers material from previous lectures and graded projects



Enrollment

- Additional class permissions being issued
- Both sections (498 & 599)
- If you haven't received a class permission come see Anthony after lecture





Linear Classifiers



Building Block of Neural Networks

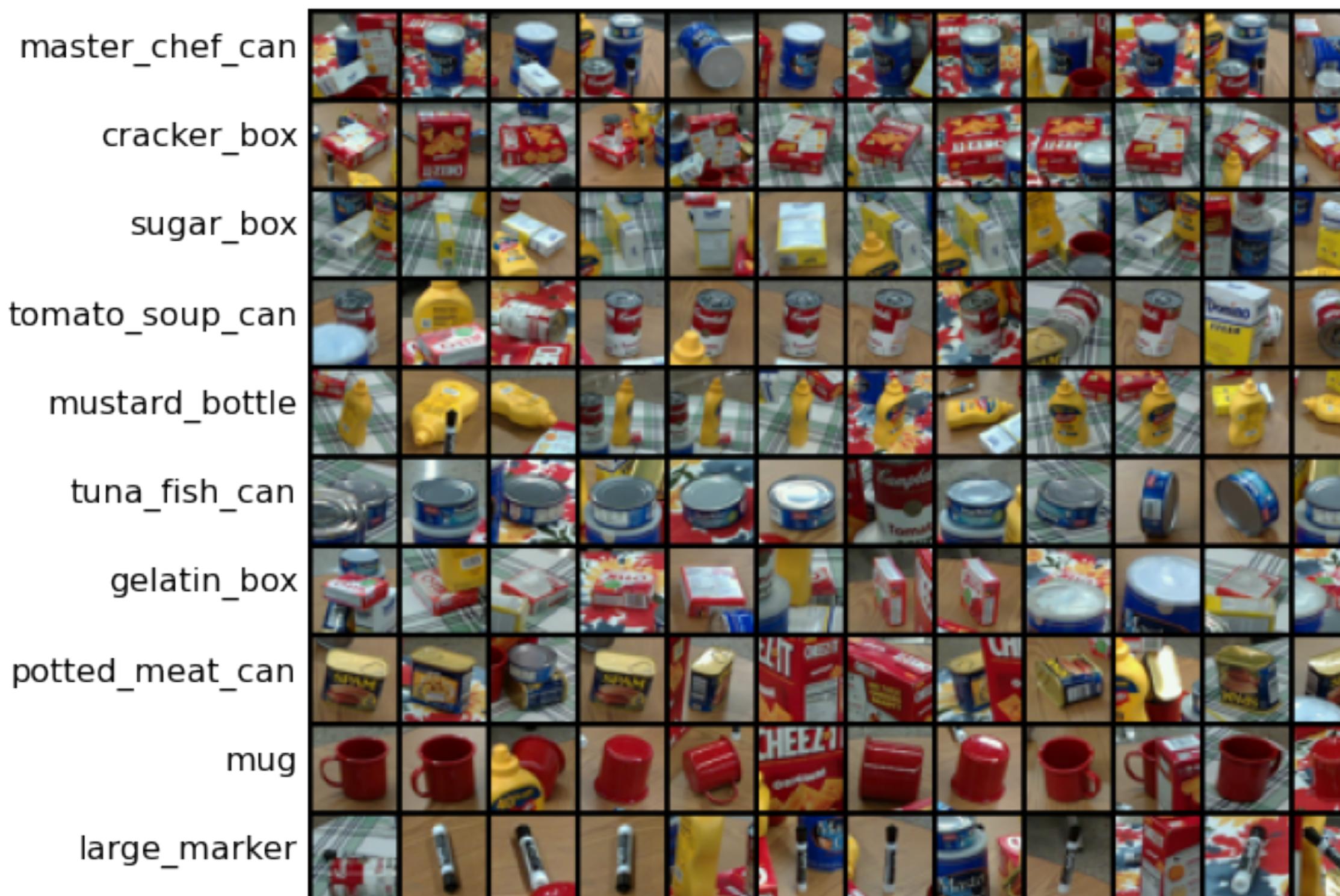
Linear
classifiers



[This image is CC0 1.0 public domain](#)

Recall PROPS

Progress Robot Object Perception Samples Dataset



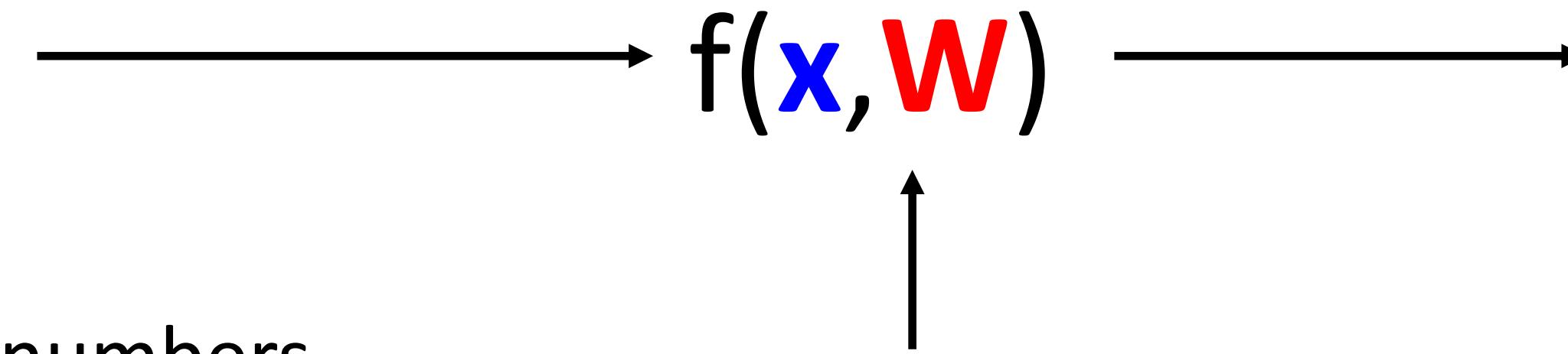
10 classes
32x32 RGB images
50k training images (5k per class)
10k test images (1k per class)

Chen et al., “ProgressLabeller: Visual Data Stream Annotation for Training Object-Centric 3D Perception”, IROS, 2022.

Parametric Approach



Array of **32x32x3** numbers
(3072 numbers total)



\mathbf{W}
parameters
or weights

Parametric Approach – Linear Classifier

Image



Array of **32x32x3** numbers
(3072 numbers total)

$$f(x, W) = Wx$$

$$f(\textcolor{blue}{x}, \textcolor{red}{W})$$



W
parameters
or weights

10 numbers giving
class scores

Parametric Approach – Linear Classifier

Image



Array of **32x32x3** numbers
(3072 numbers total)

$$f(x, W) = \boxed{W} \boxed{x}$$

(10,) (10, 3072)

10 numbers giving
class scores

W
parameters
or weights

Parametric Approach – Linear Classifier

Image



Array of **32x32x3** numbers
(3072 numbers total)

$$f(x, W) = \boxed{W} \boxed{x} + \boxed{b}$$

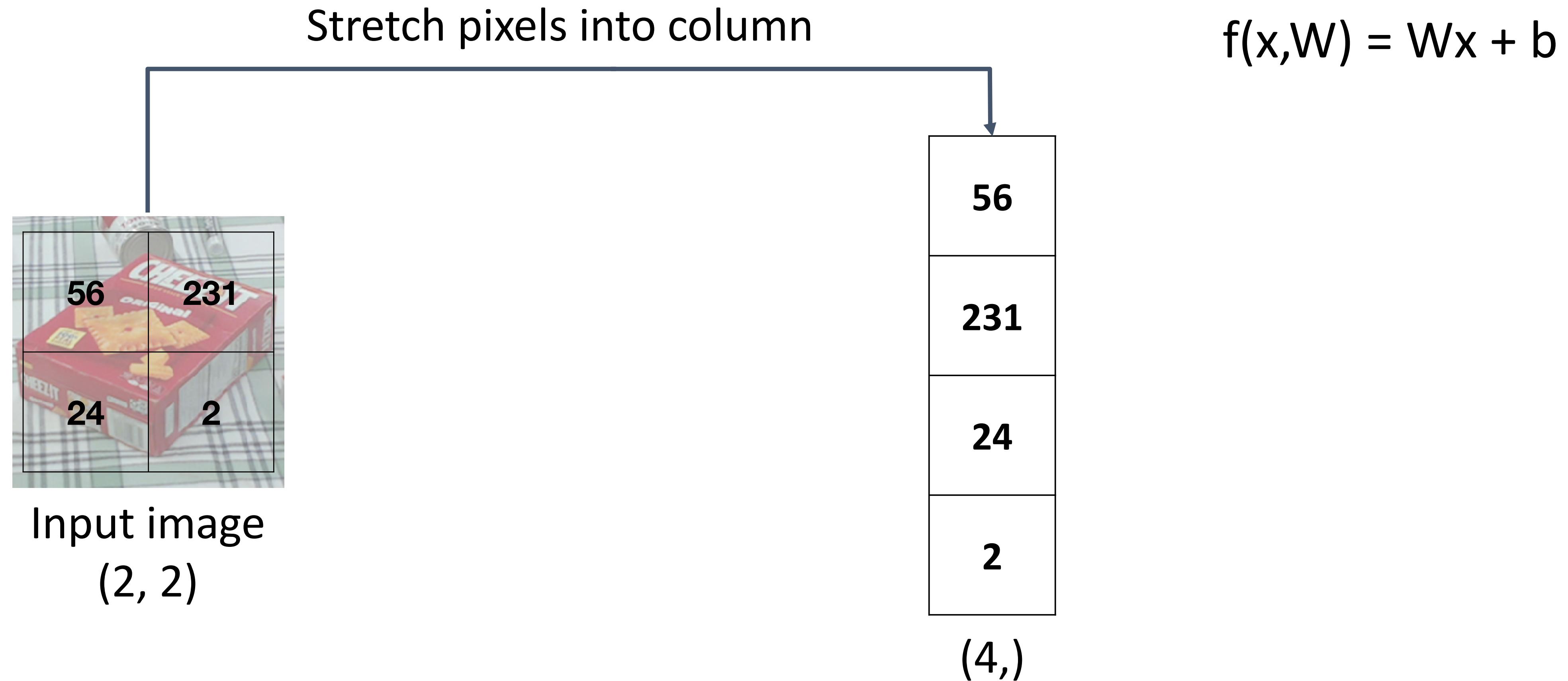
(10,) (10, 3072)

W

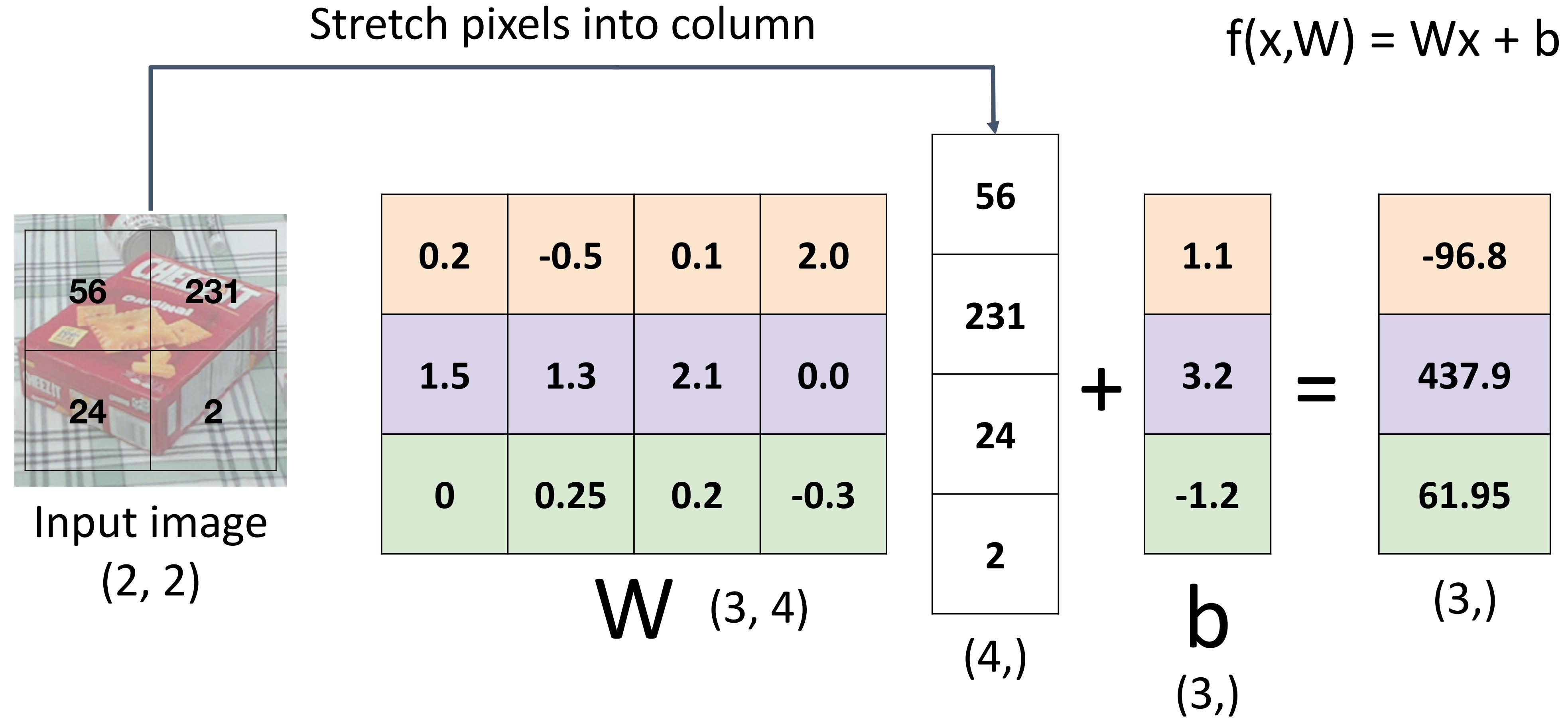
parameters
or weights

10 numbers giving
class scores

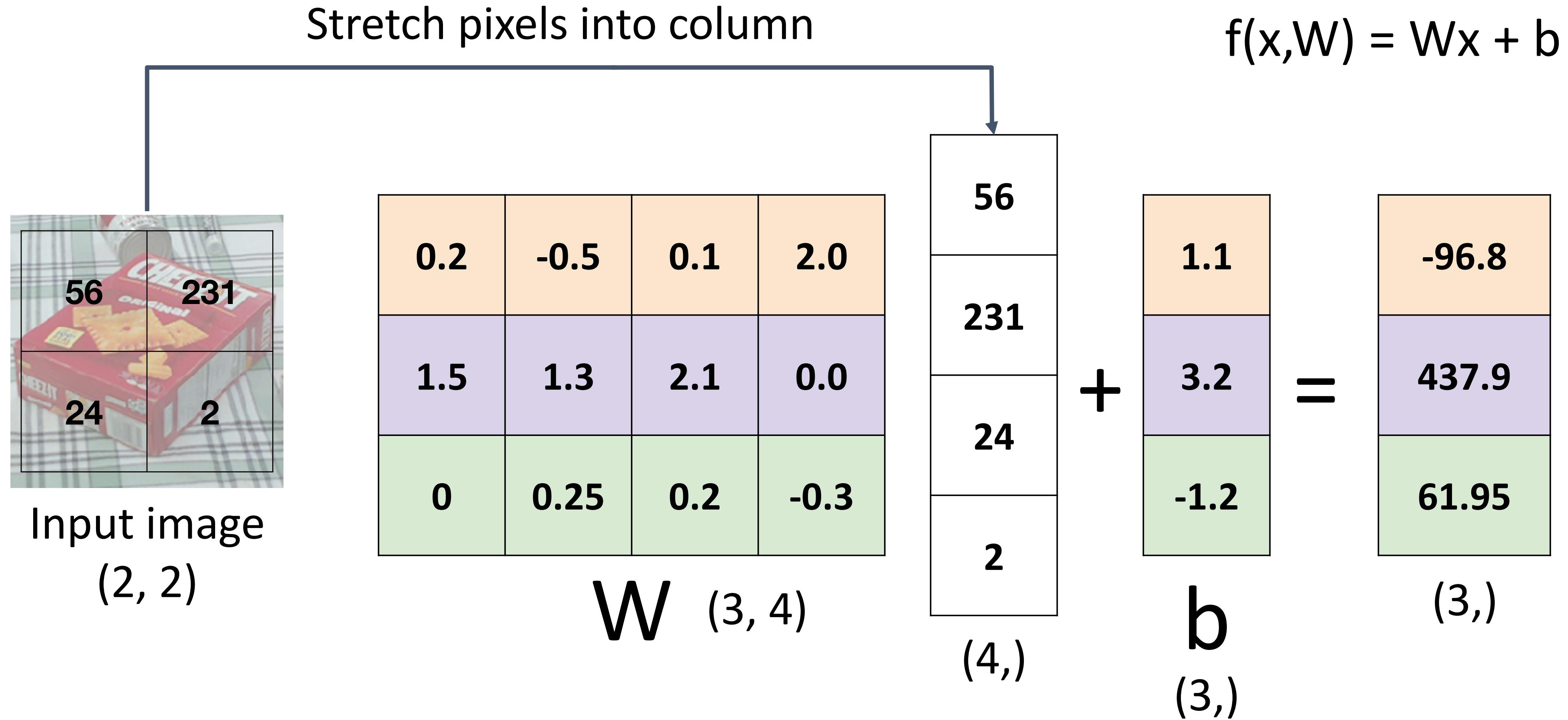
Example for 2x2 Image, 3 classes (crackers/mug/sugar)



Example for 2x2 Image, 3 classes (crackers/mug/sugar)

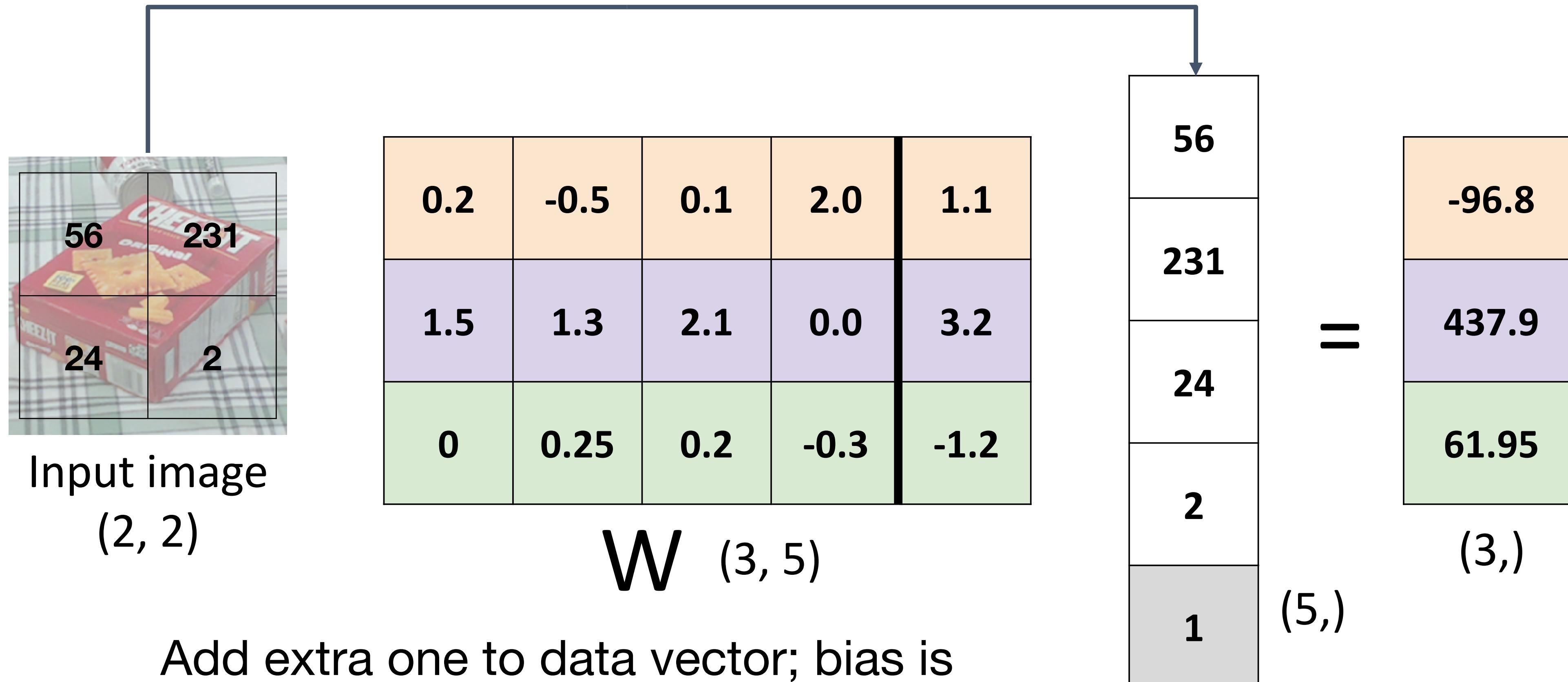


Linear Classifier—Algebraic Viewpoint



Linear Classifier—Bias Trick

Stretch pixels into column





Linear Classifier—Predictions are Linear

$$f(x, W) = Wx \quad (\text{ignore bias})$$

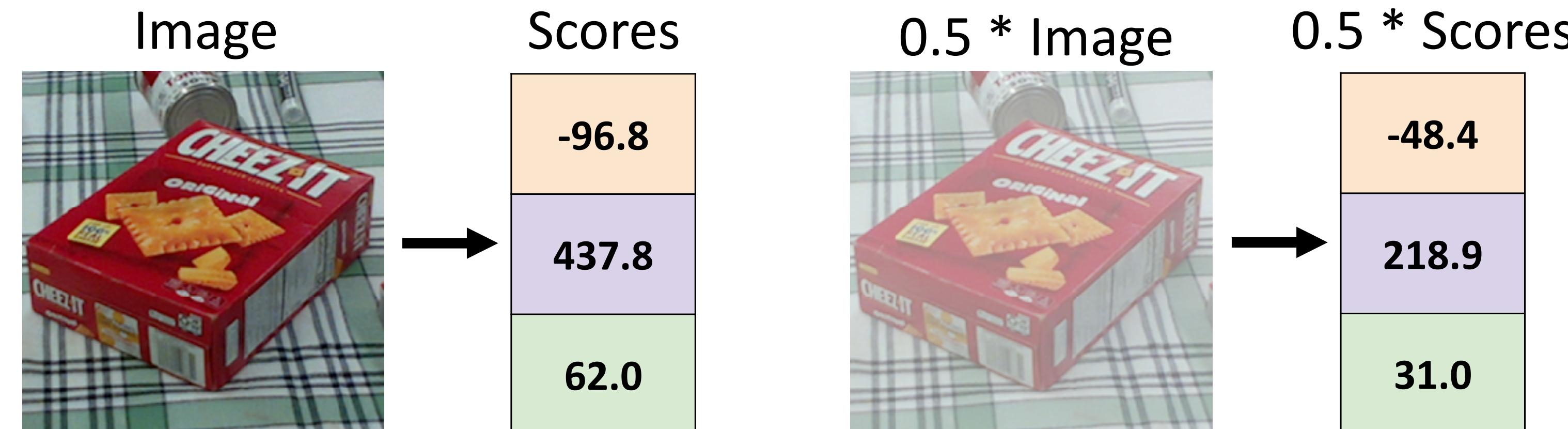
$$f(cx, W) = W(cx) = c * f(x, W)$$



Linear Classifier—Predictions are Linear

$$f(x, W) = Wx \quad (\text{ignore bias})$$

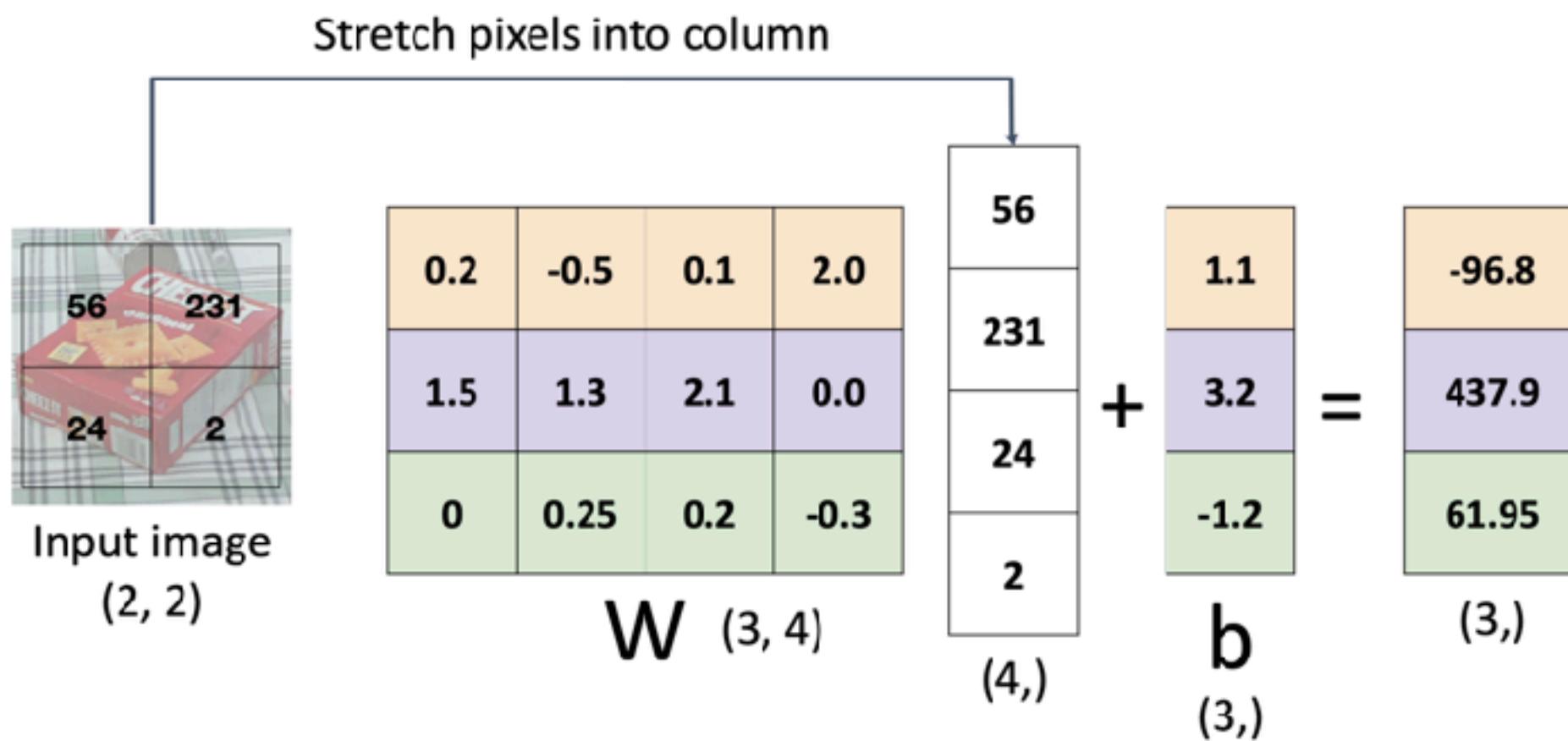
$$f(cx, W) = W(cx) = c * f(x, W)$$



Interpreting a Linear Classifier

Algebraic Viewpoint

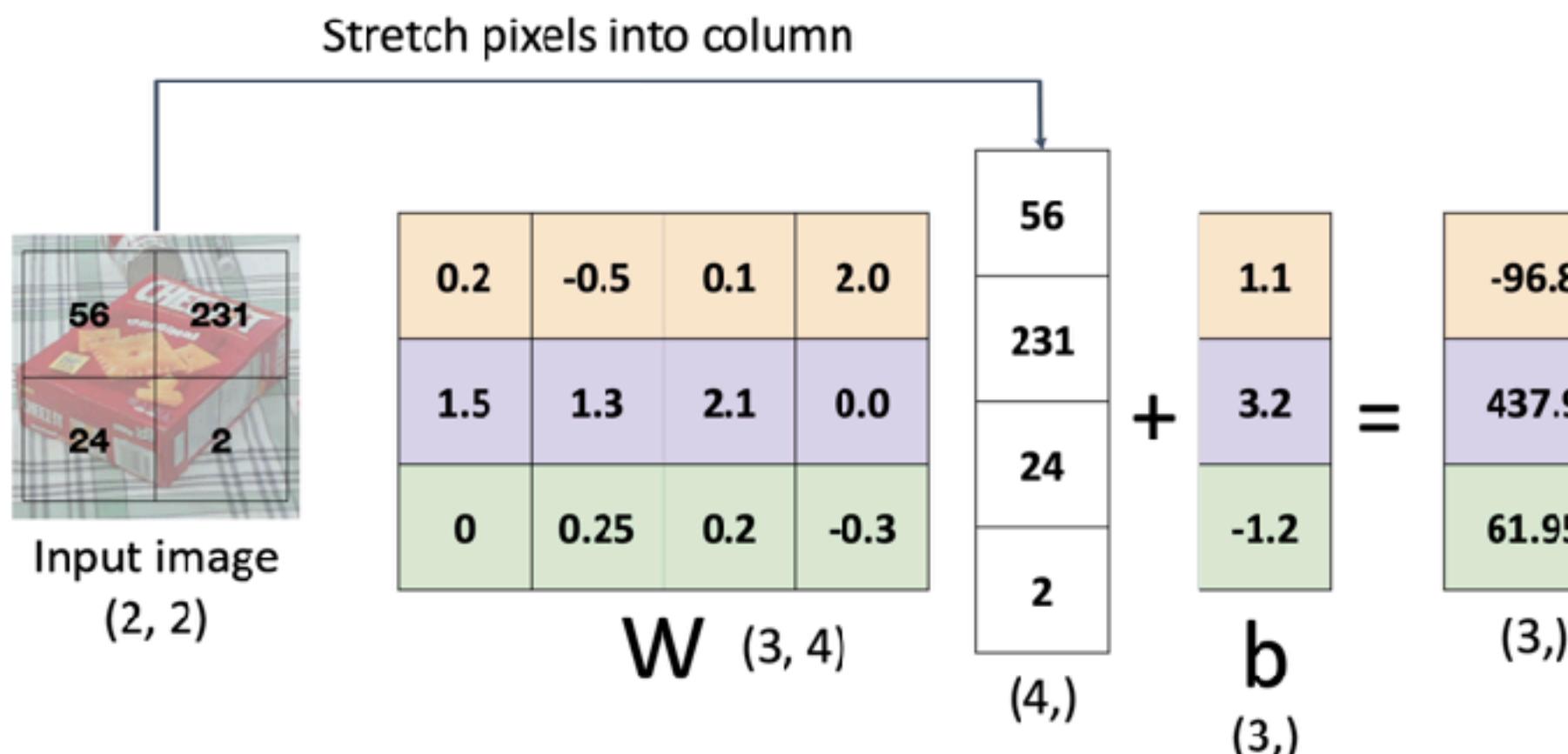
$$f(x, W) = Wx + b$$



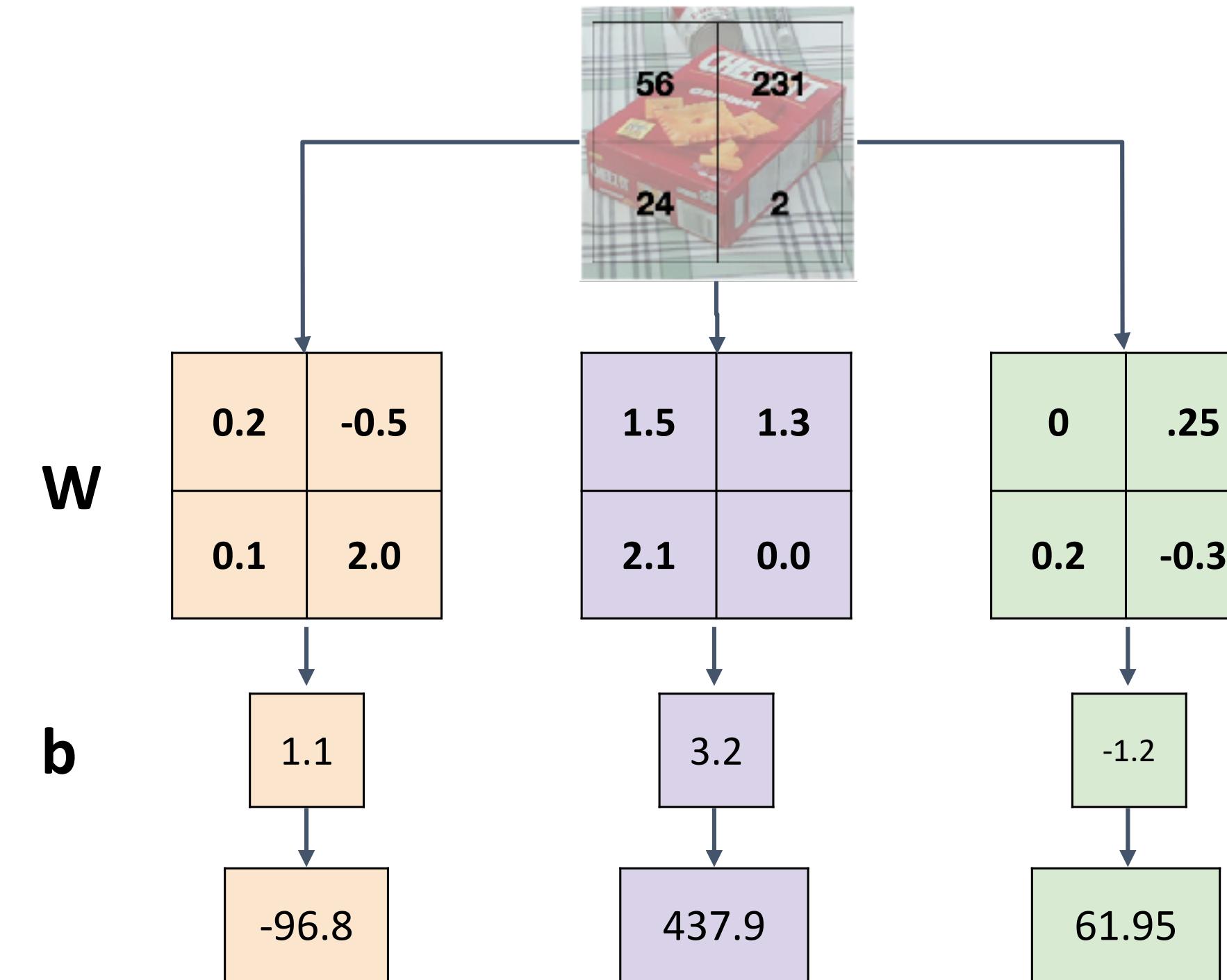
Interpreting a Linear Classifier

Algebraic Viewpoint

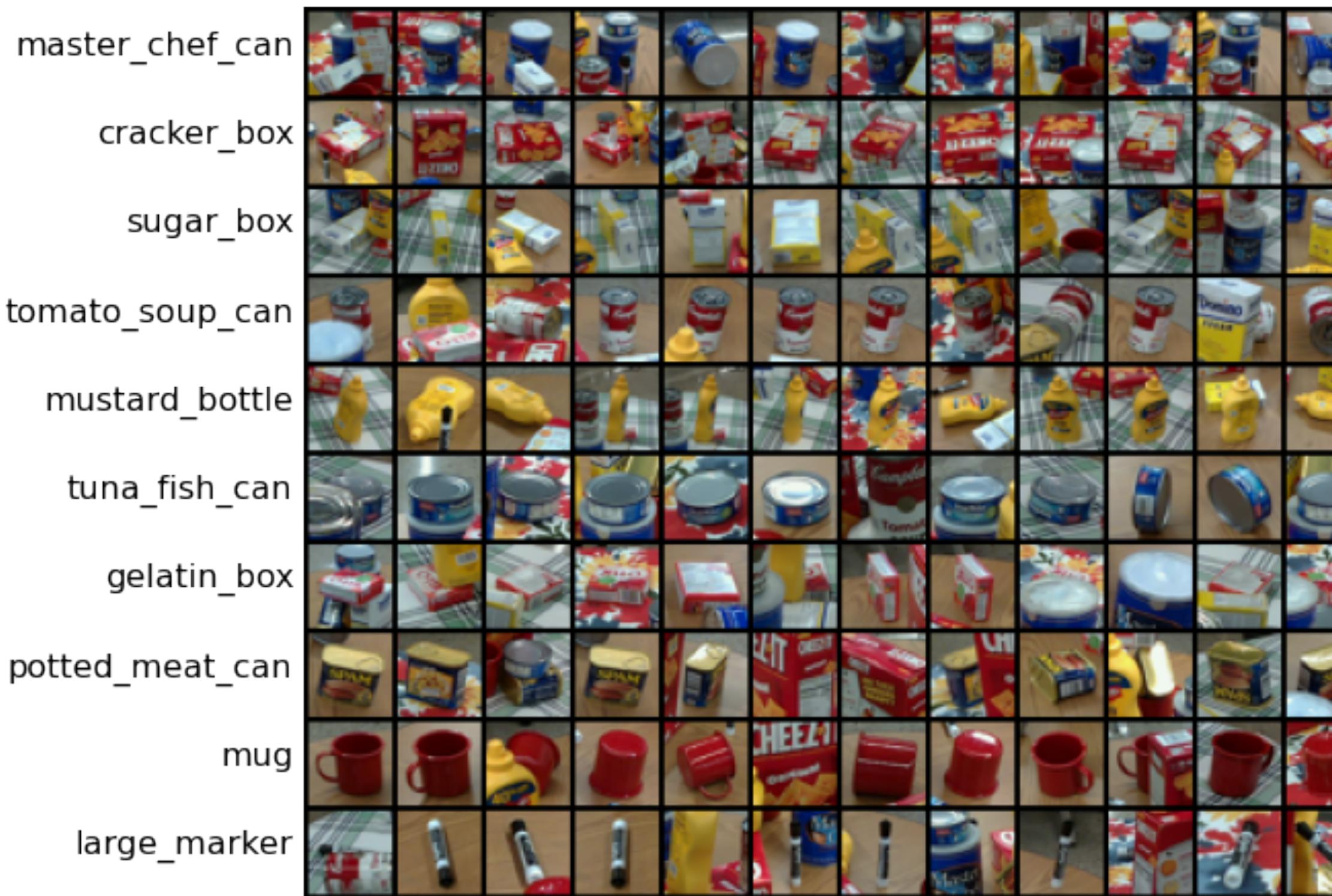
$$f(x, W) = Wx + b$$



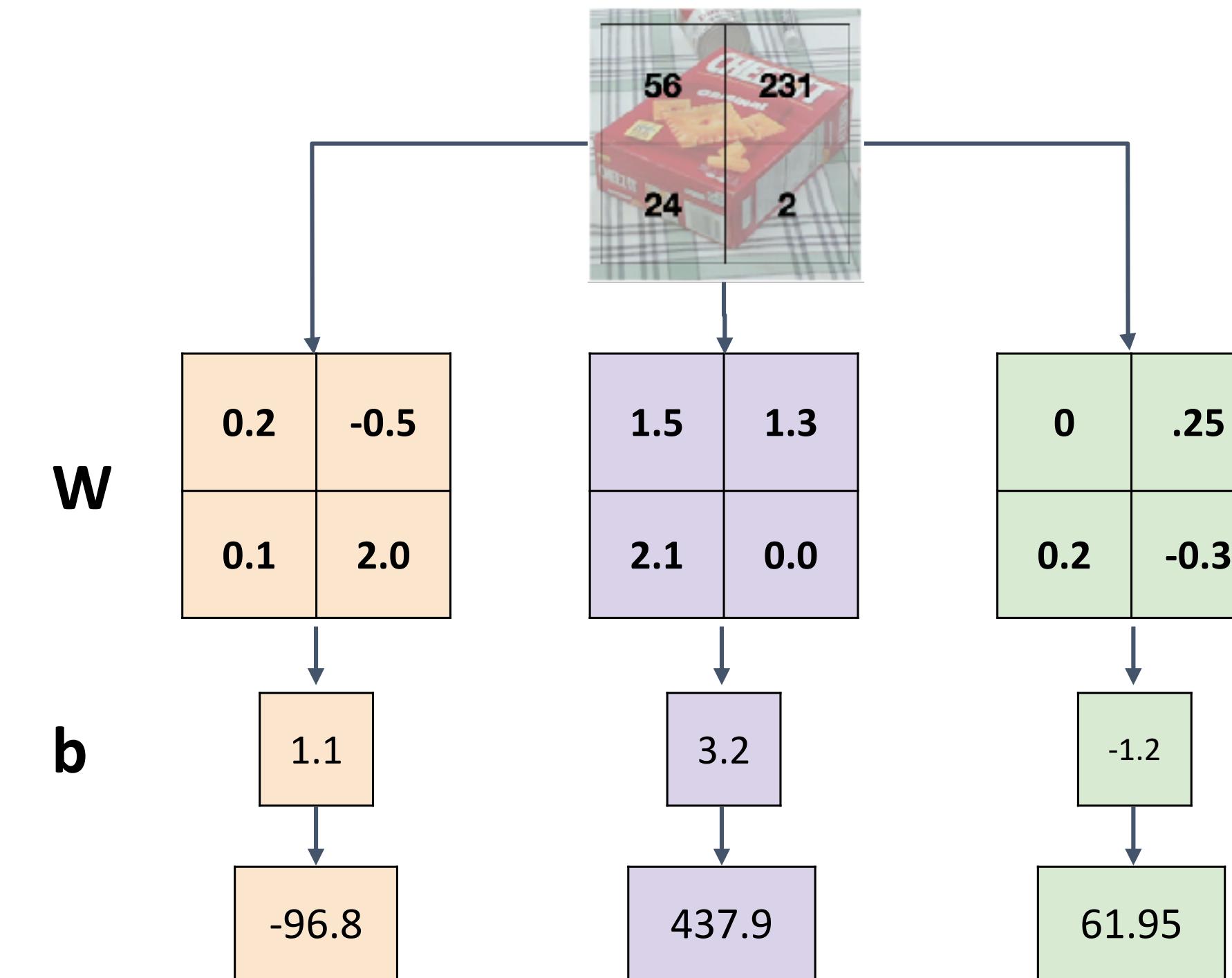
Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



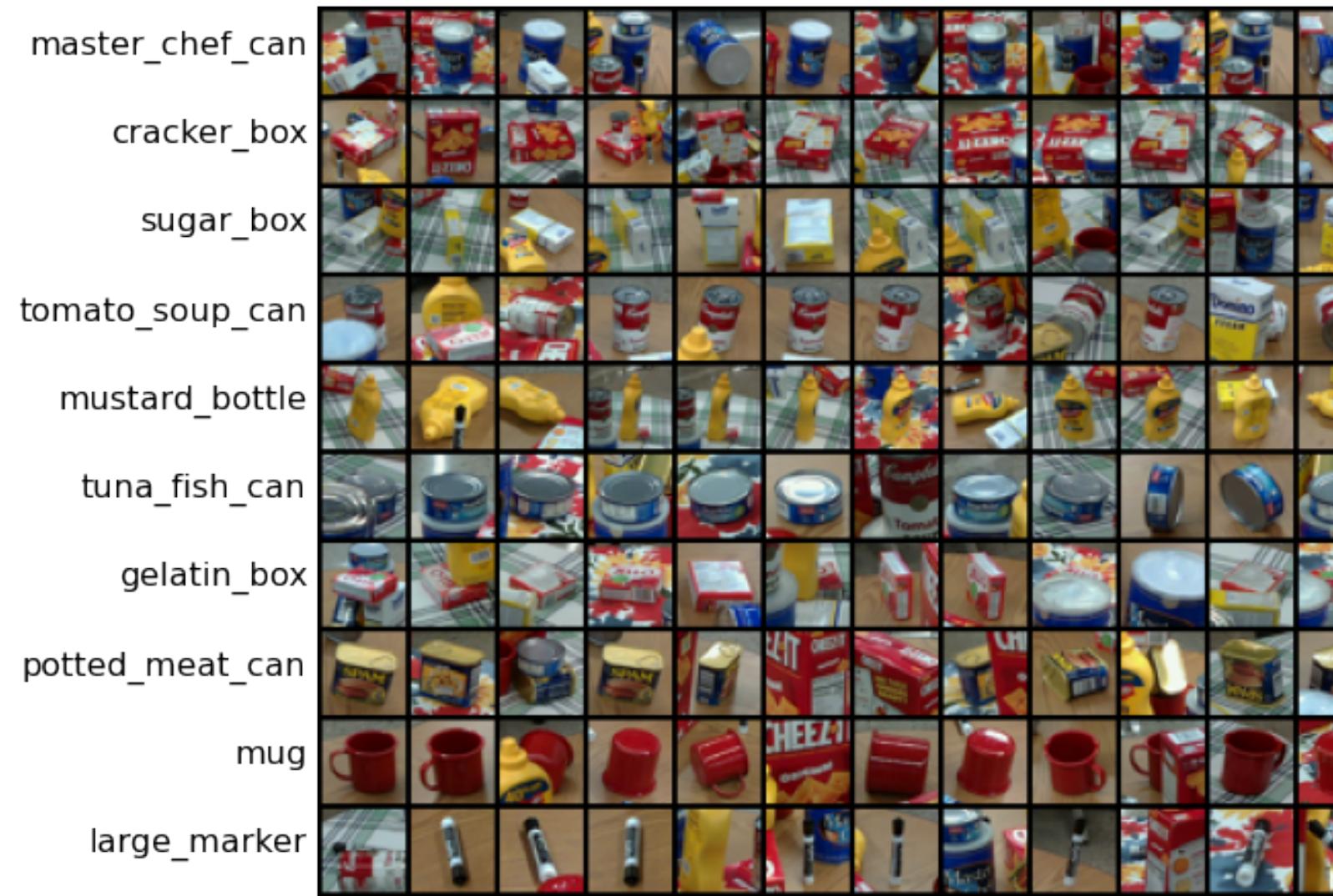
Interpreting a Linear Classifier



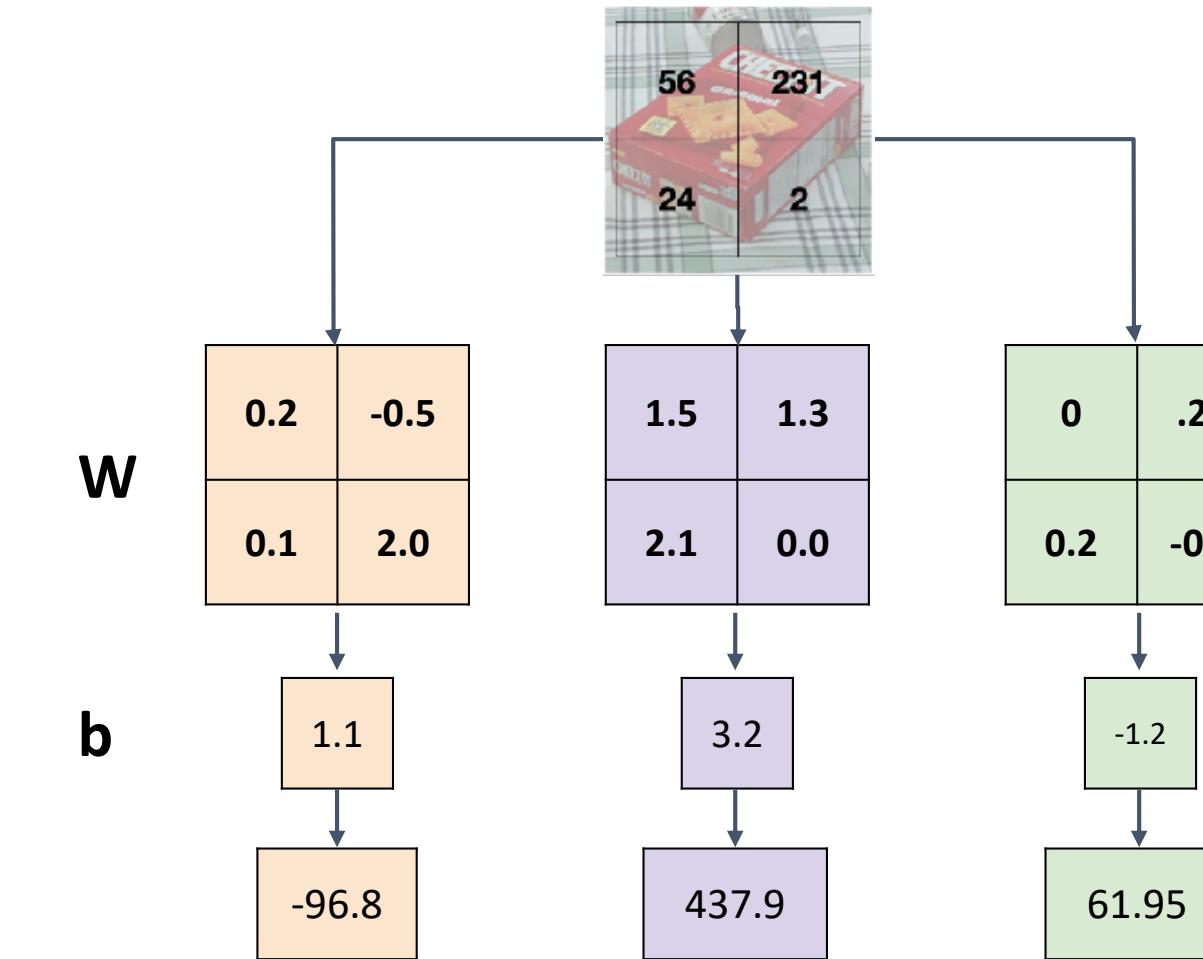
Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



Interpreting a Linear Classifier



Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!

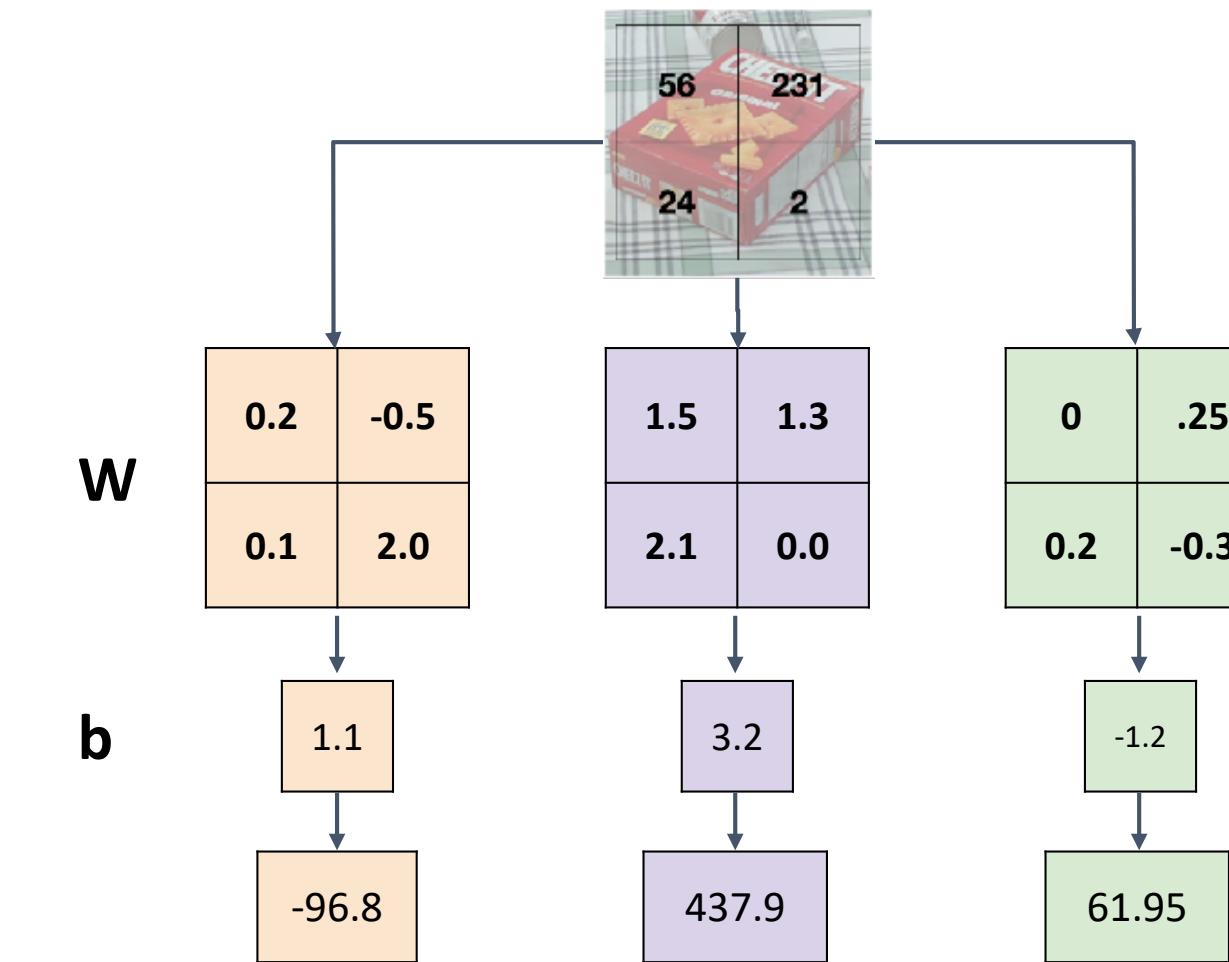


Interpreting a Linear Classifier – Visual Viewpoint

Linear classifier has one “template” per category



Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



Interpreting a Linear Classifier – Visual Viewpoint

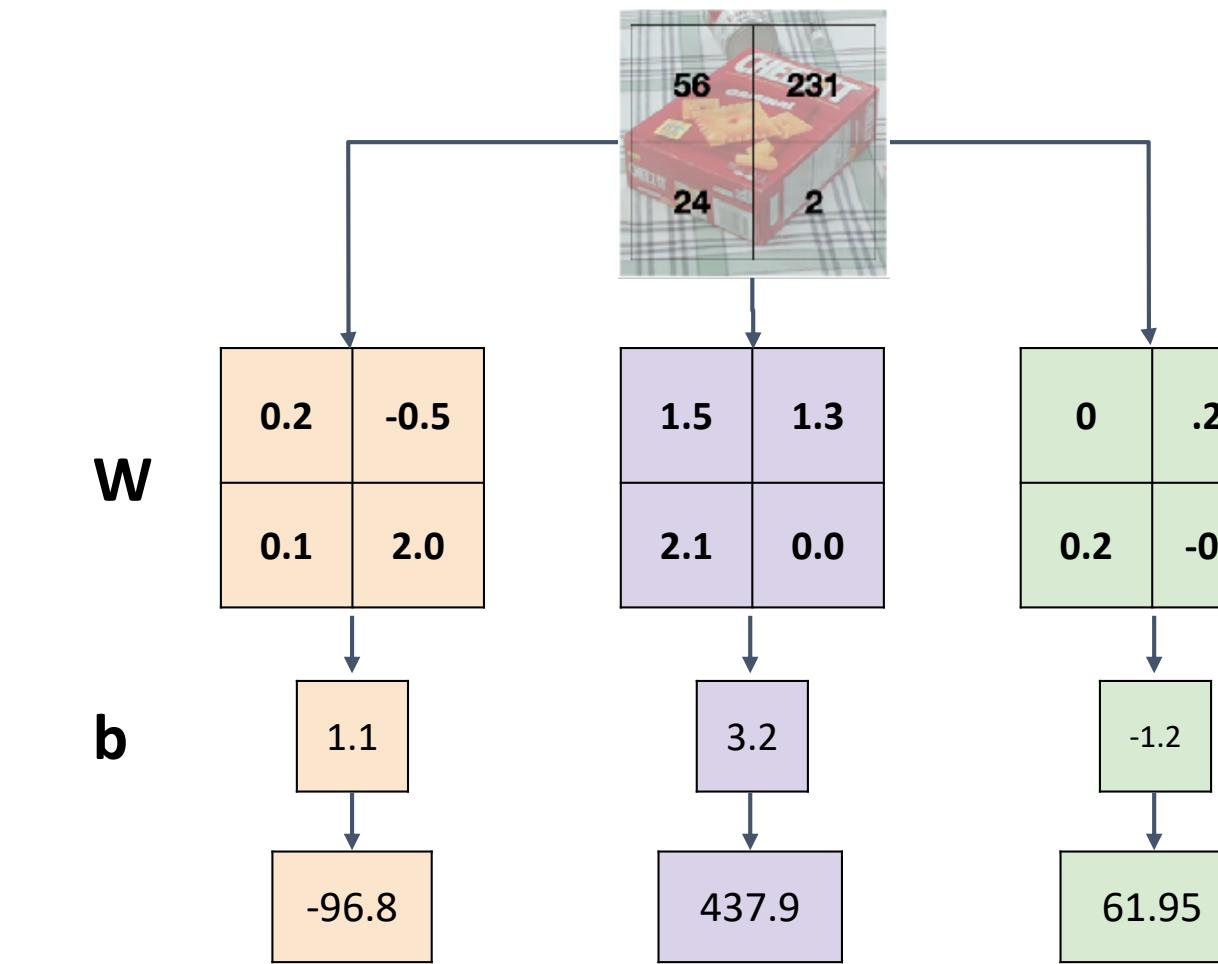
Linear classifier has one “template” per category

A single template cannot capture multiple modes of the data

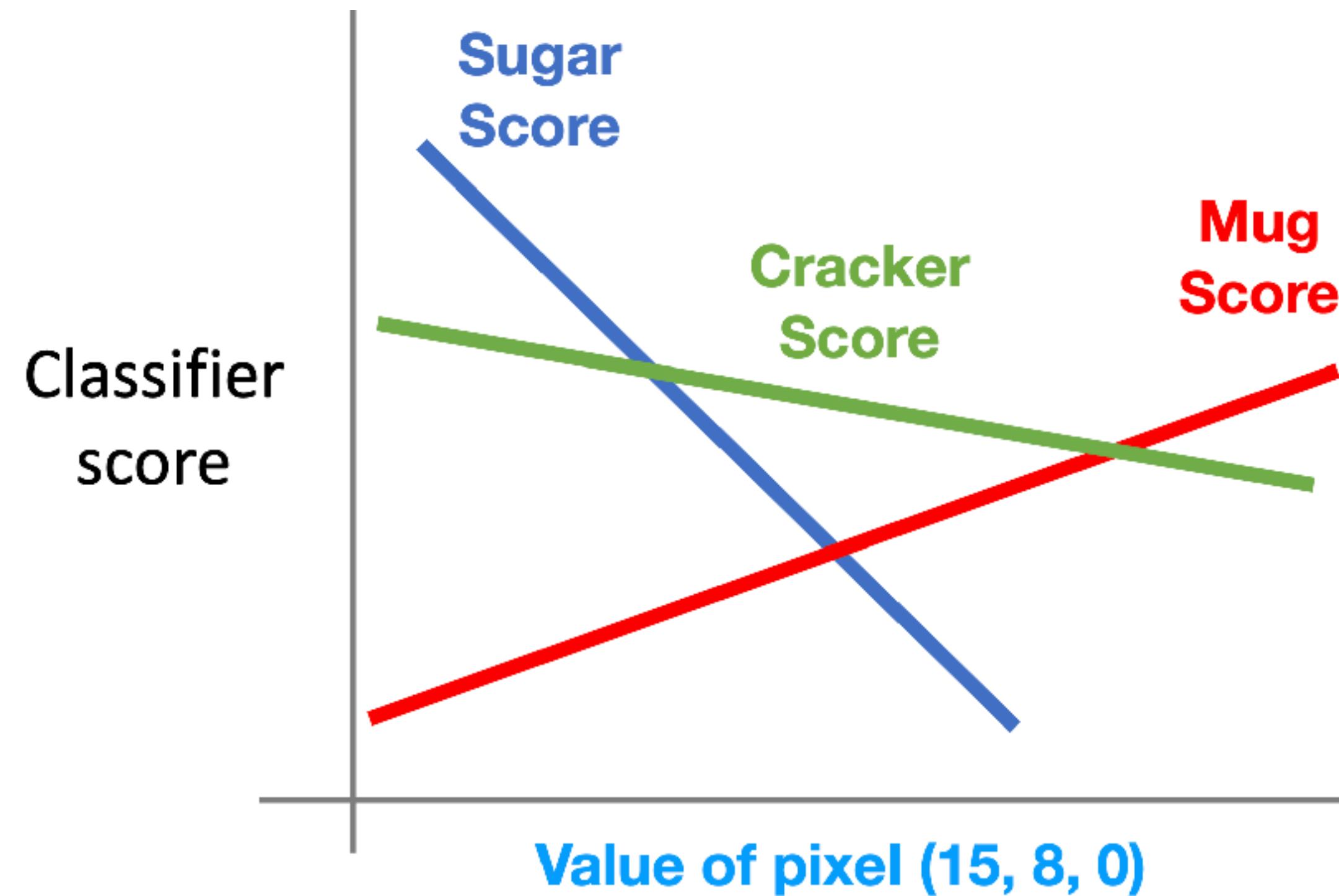
e.g. mustard bottles can rotate



Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



Interpreting a Linear Classifier—Geometric Viewpoint

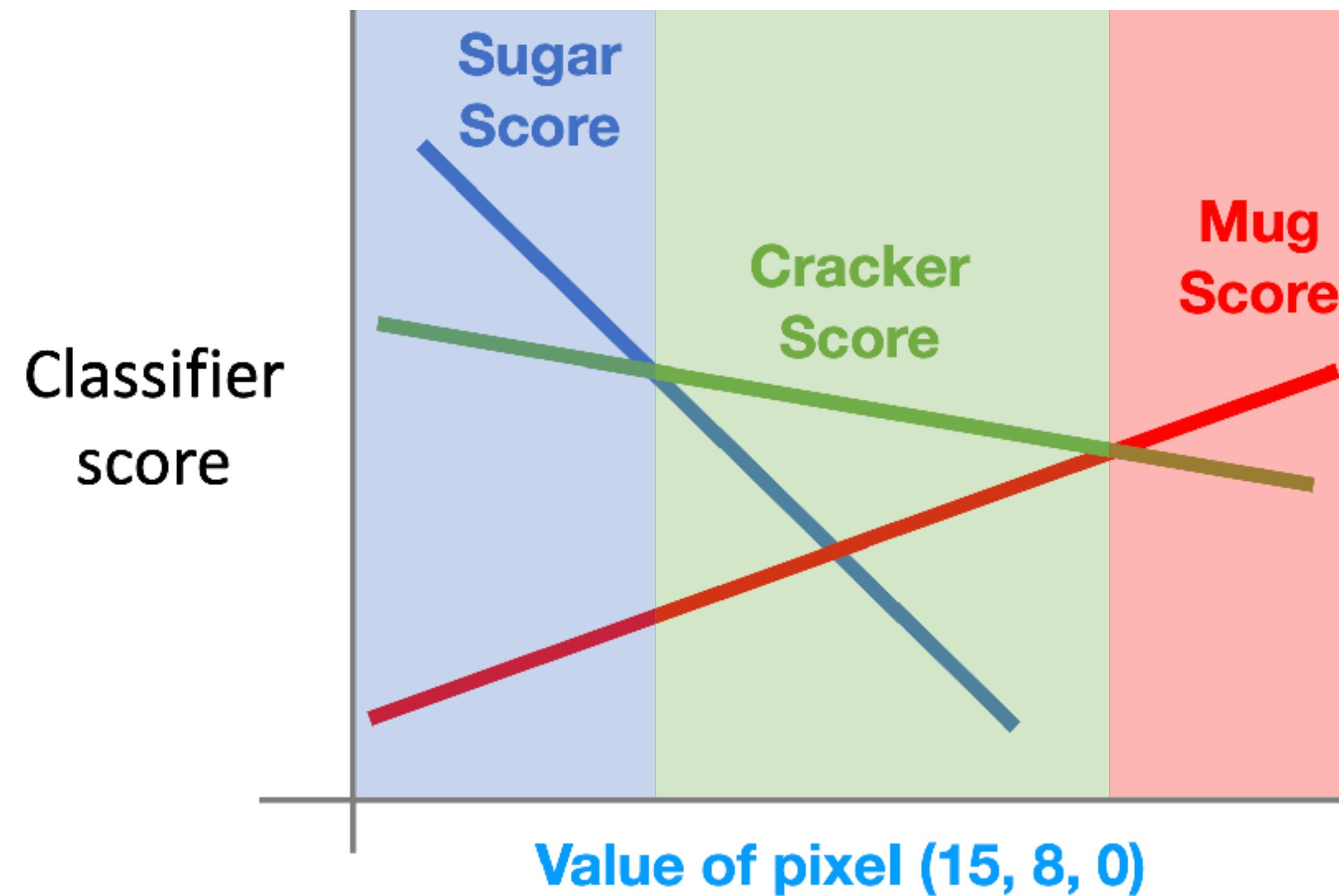


$$f(\mathbf{x}, \mathbf{W}) = \mathbf{W}\mathbf{x} + b$$



Array of **32x32x3** numbers
(3072 numbers total)

Interpreting a Linear Classifier—Geometric Viewpoint

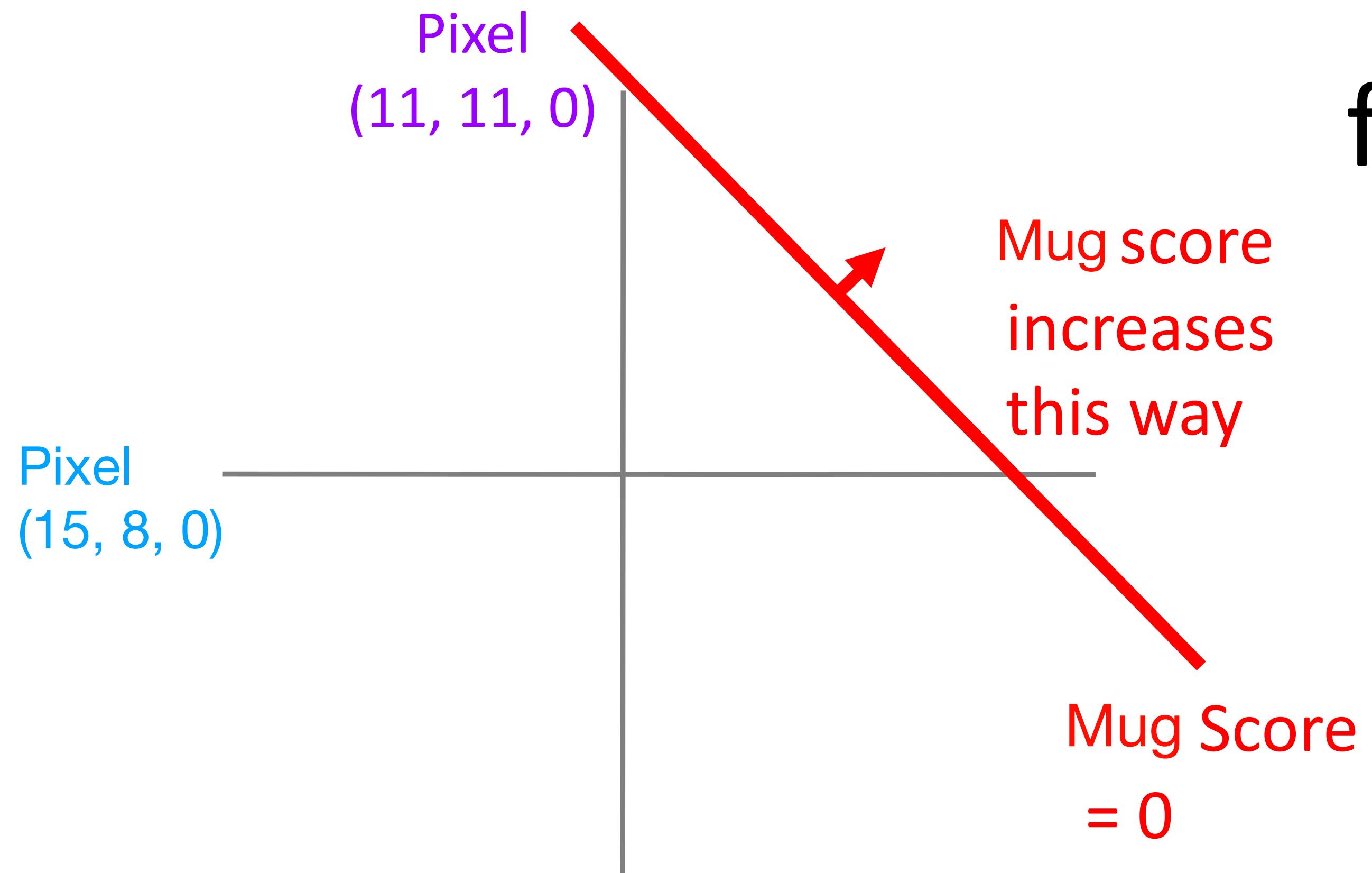


$$f(\mathbf{x}, \mathbf{W}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$



Array of **32x32x3** numbers
(3072 numbers total)

Interpreting a Linear Classifier—Geometric Viewpoint

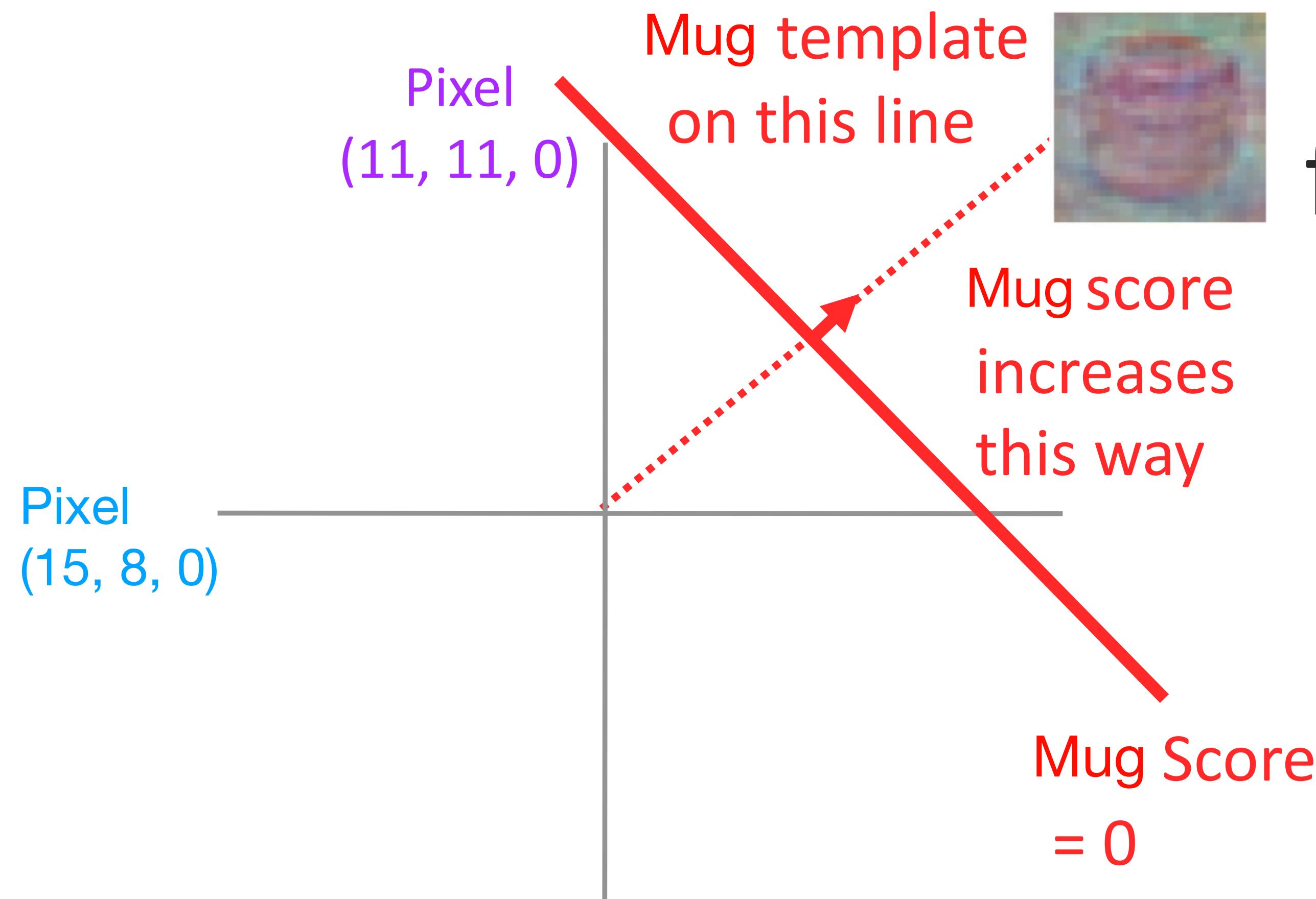


$$f(x, W) = Wx + b$$



Array of **32x32x3** numbers
(3072 numbers total)

Interpreting a Linear Classifier—Geometric Viewpoint

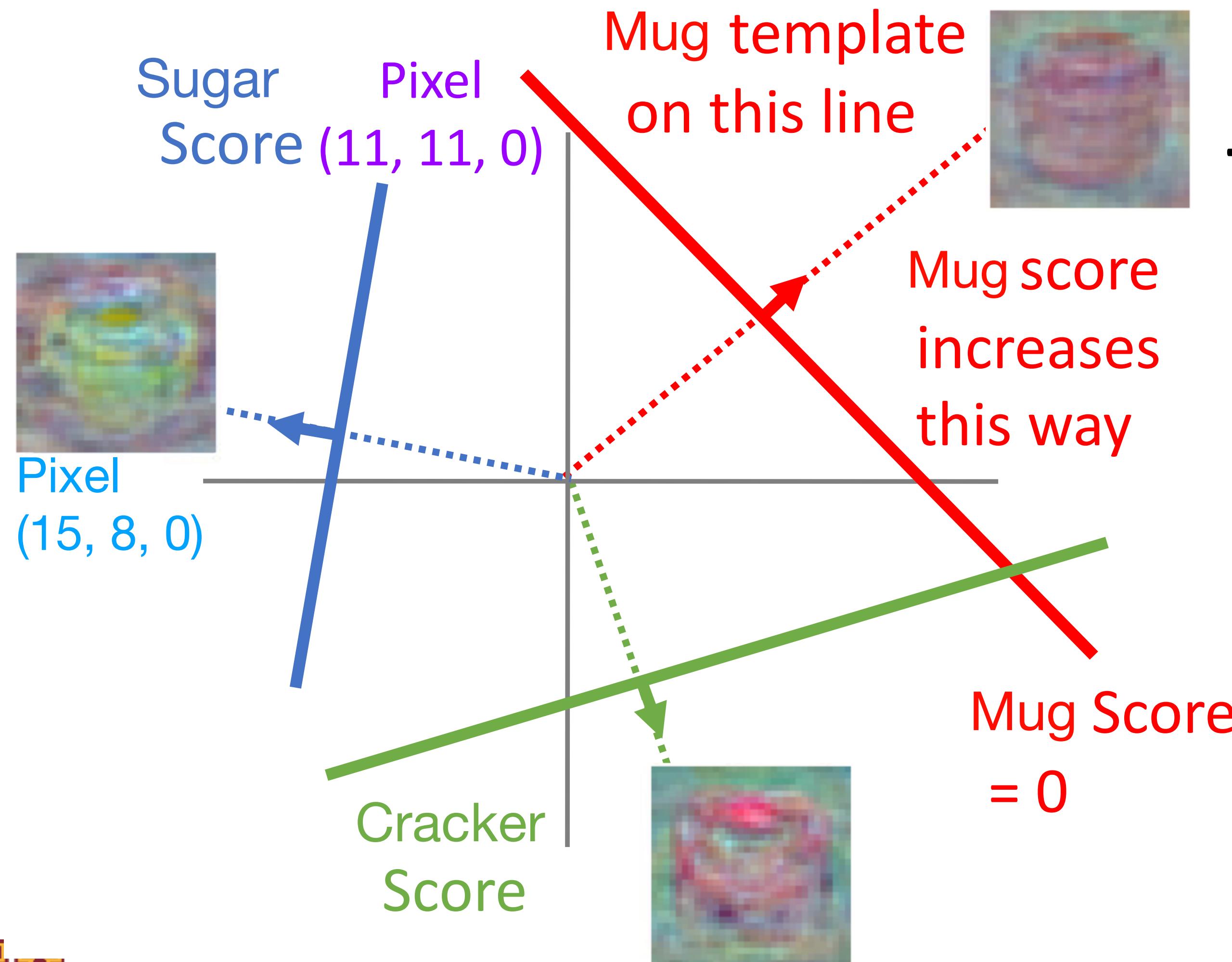


$$f(x, W) = Wx + b$$



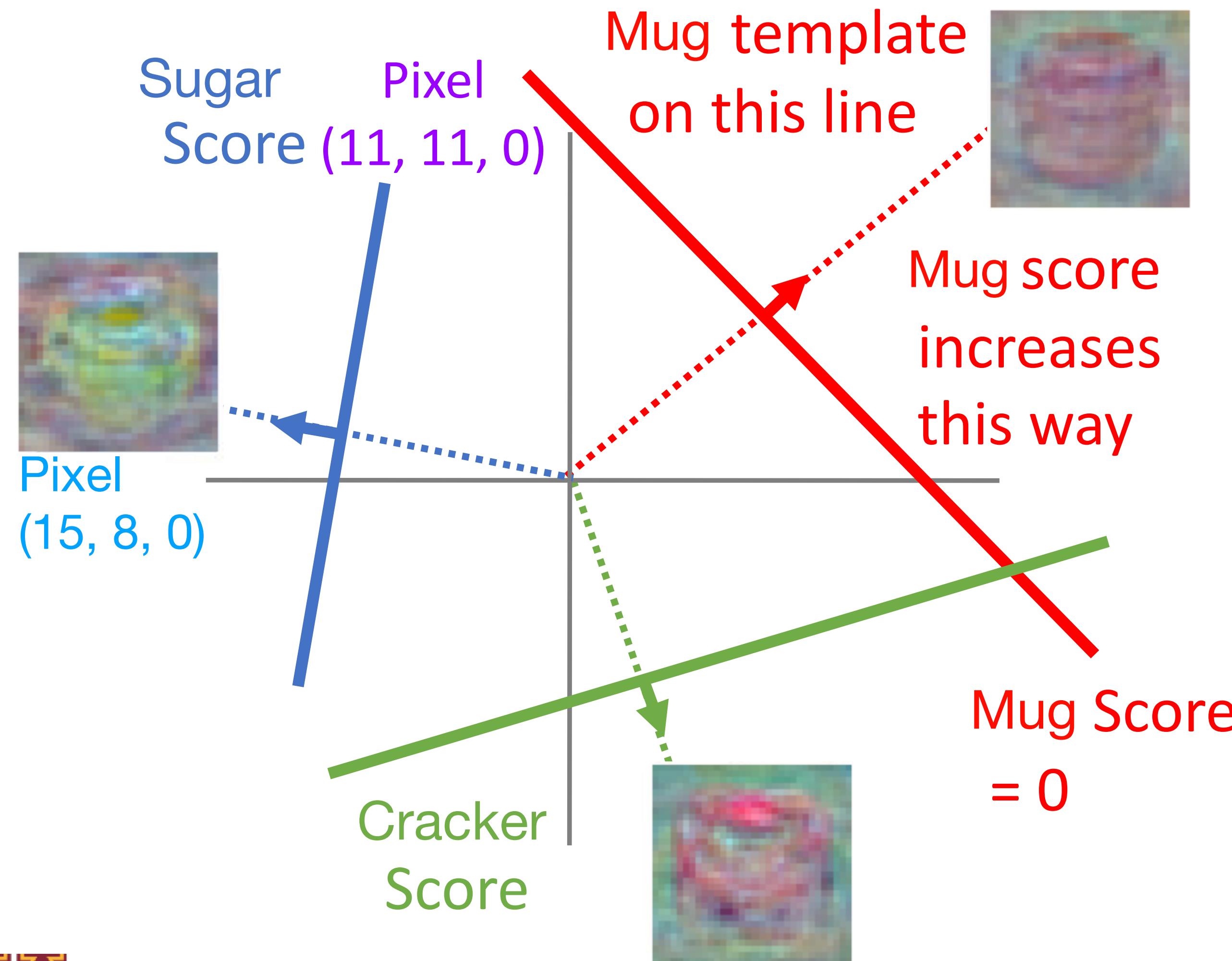
Array of **32x32x3** numbers
(3072 numbers total)

Interpreting a Linear Classifier—Geometric Viewpoint

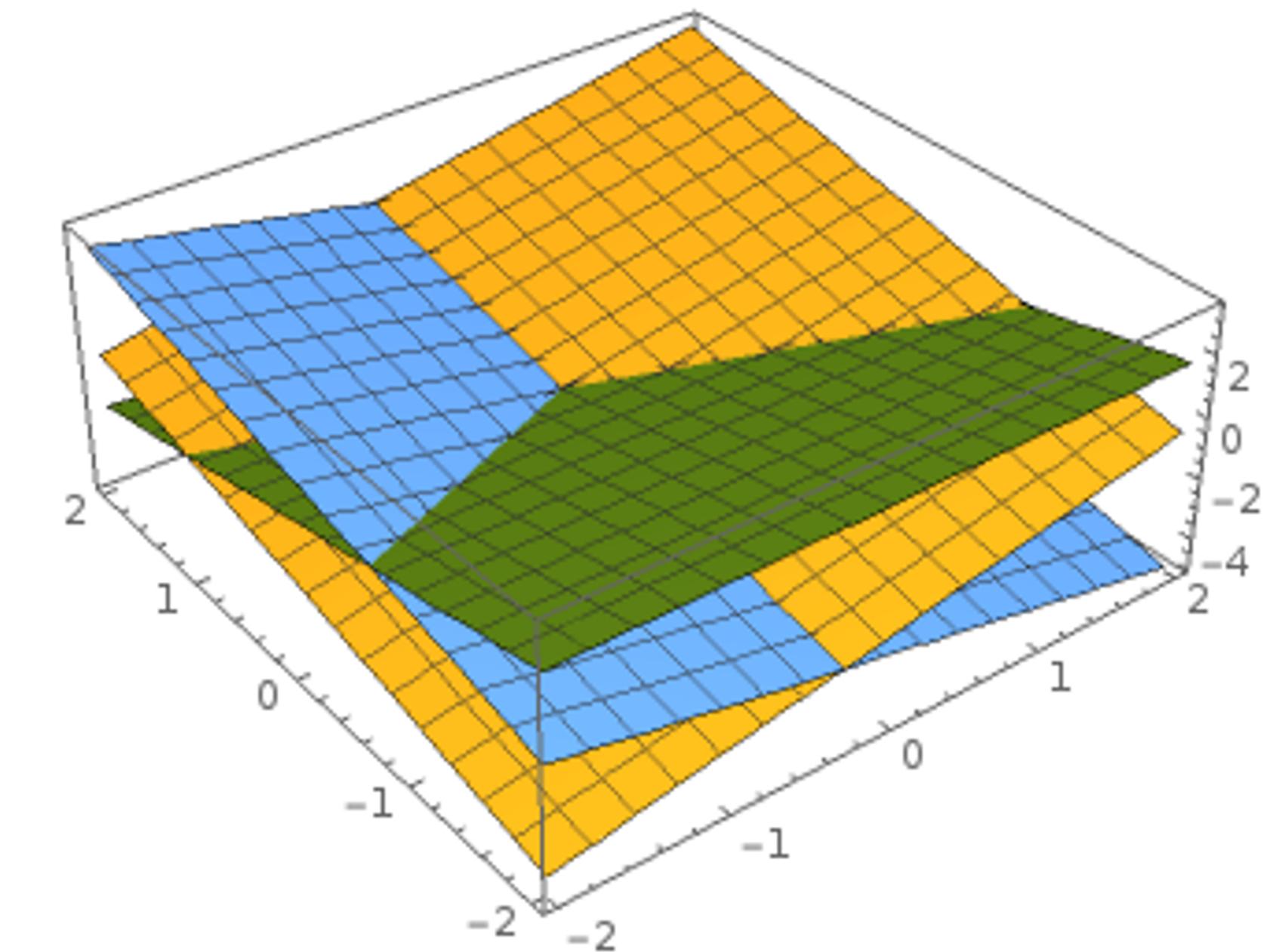


Array of **32x32x3** numbers
(3072 numbers total)

Interpreting a Linear Classifier—Geometric Viewpoint



Hyperplanes carving up a high-dimensional space

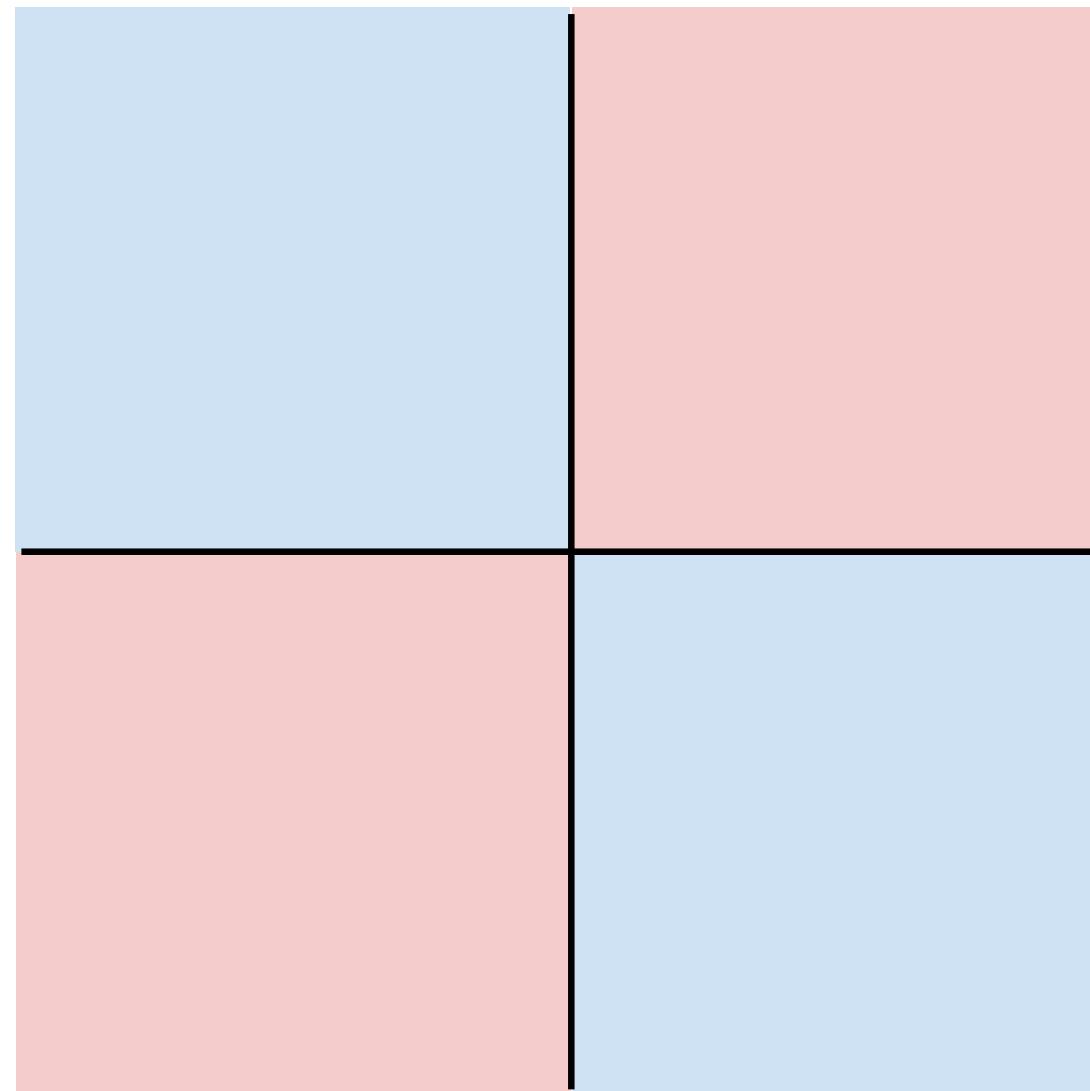


Plot created using [Wolfram Cloud](#)

Hard Cases for a Linear Classifier

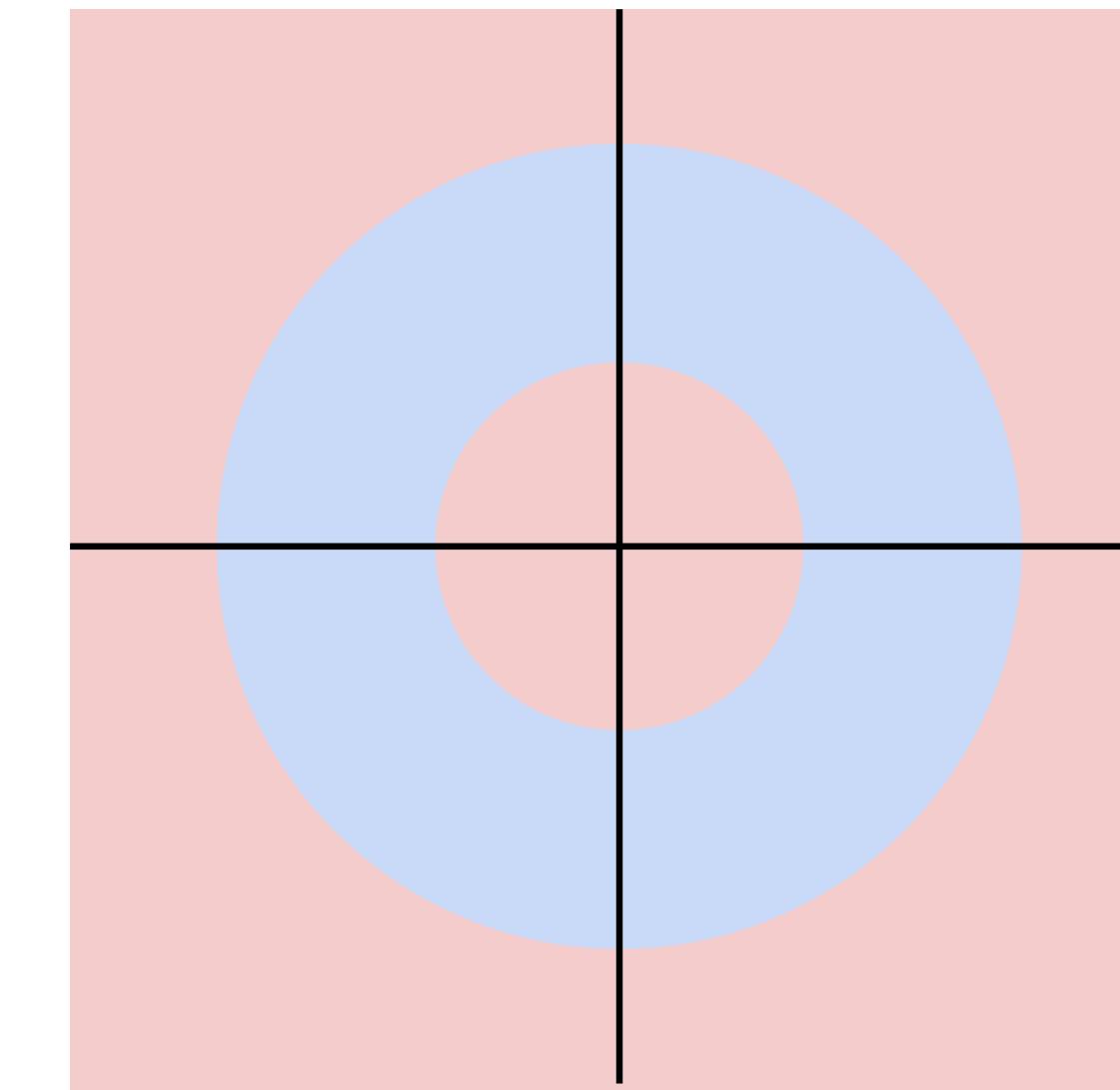
Class 1:
First and third quadrants

Class 2:
Second and fourth quadrants



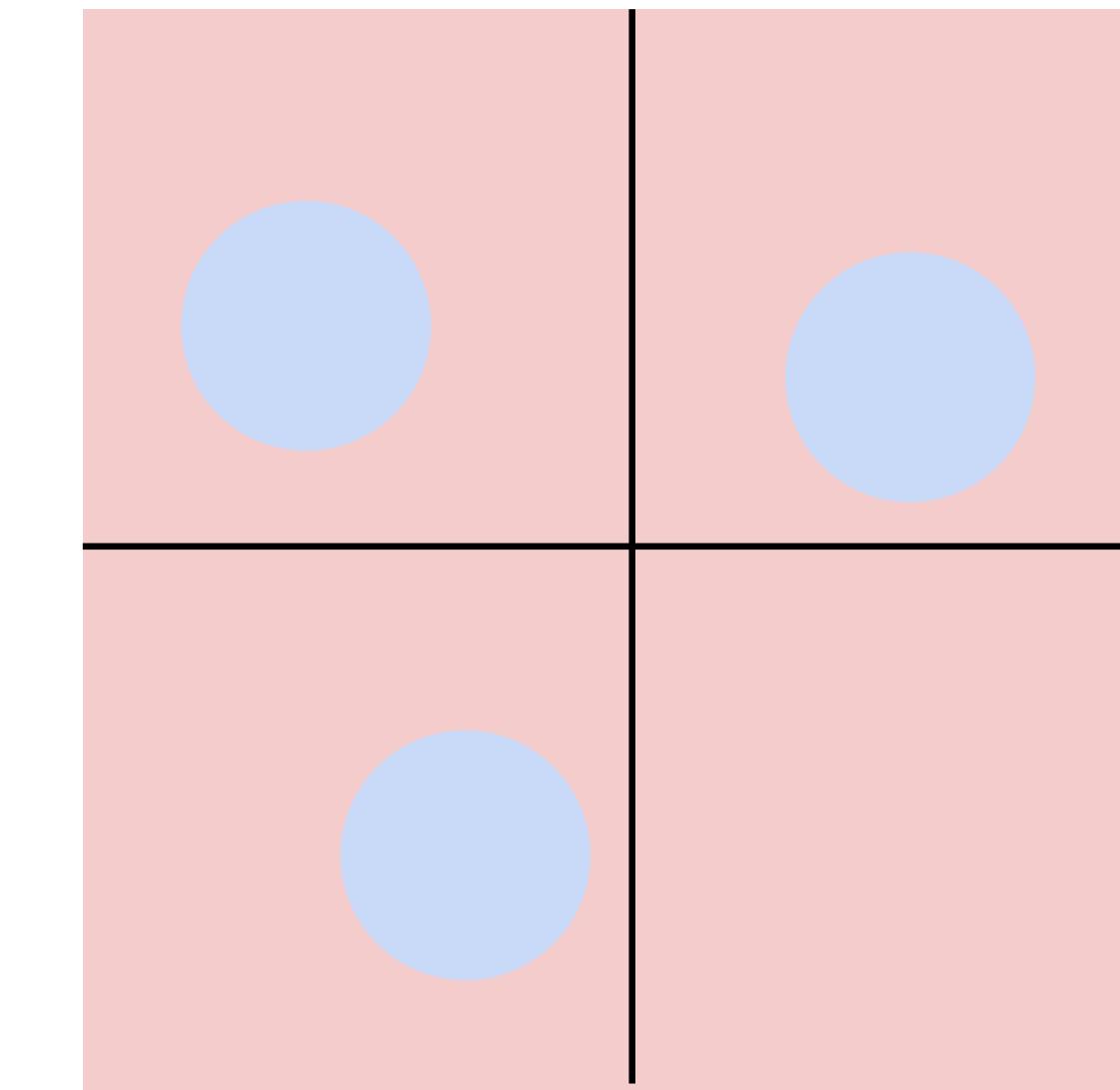
Class 1:
 $1 \leq L_2 \text{ norm} \leq 2$

Class 2:
Everything else



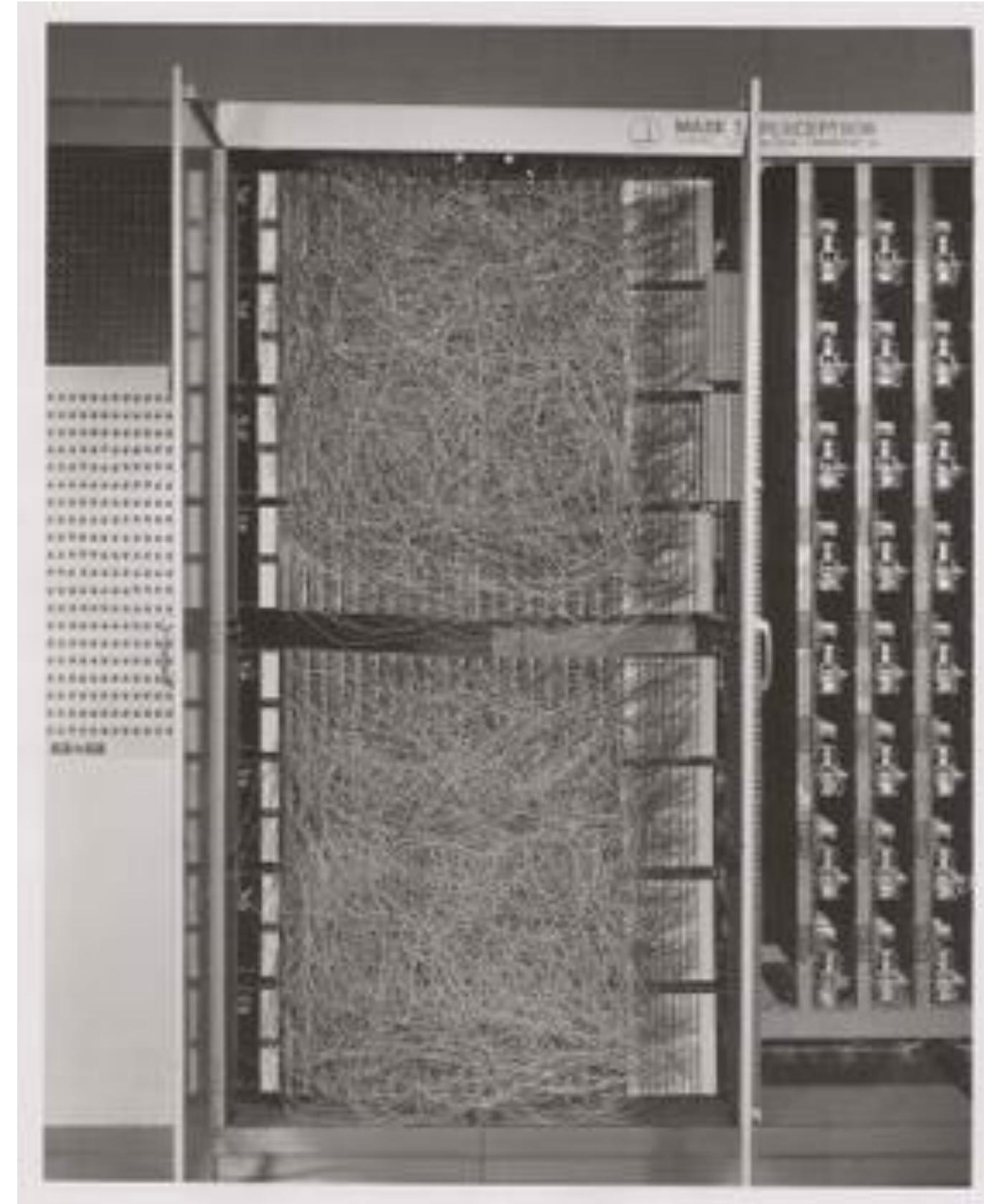
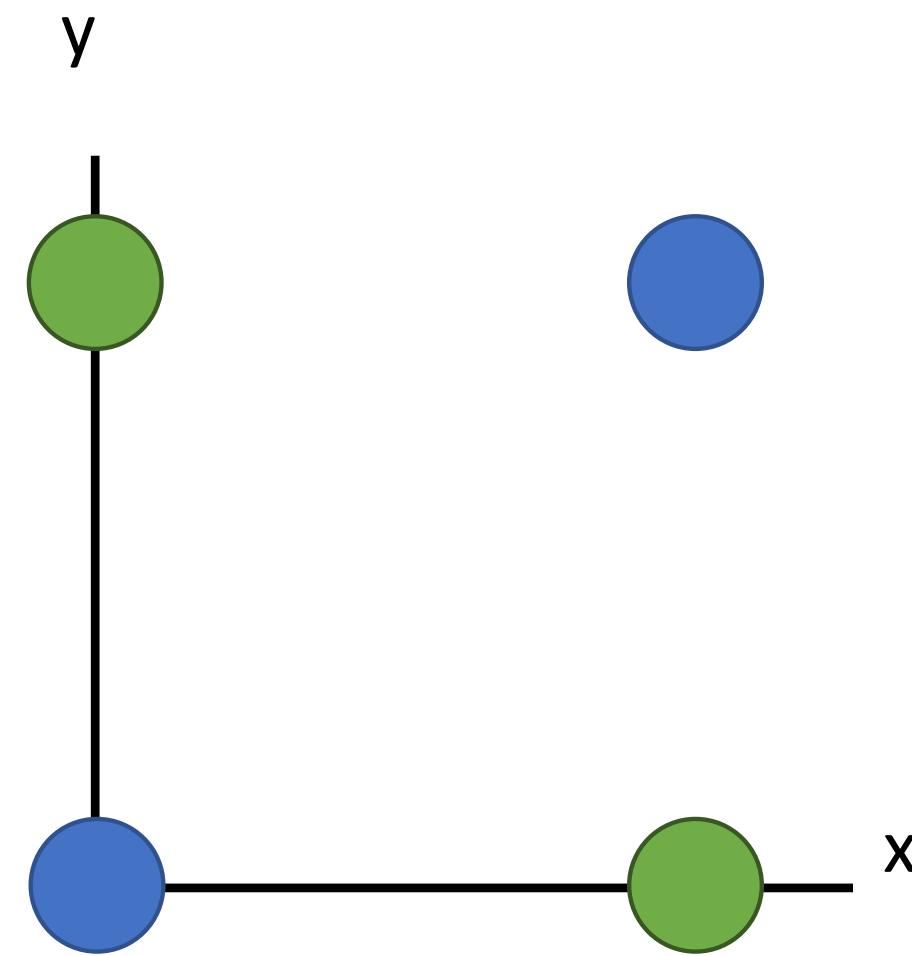
Class 1:
Three modes

Class 2:
Everything else



Perceptron Couldn't Learn XOR

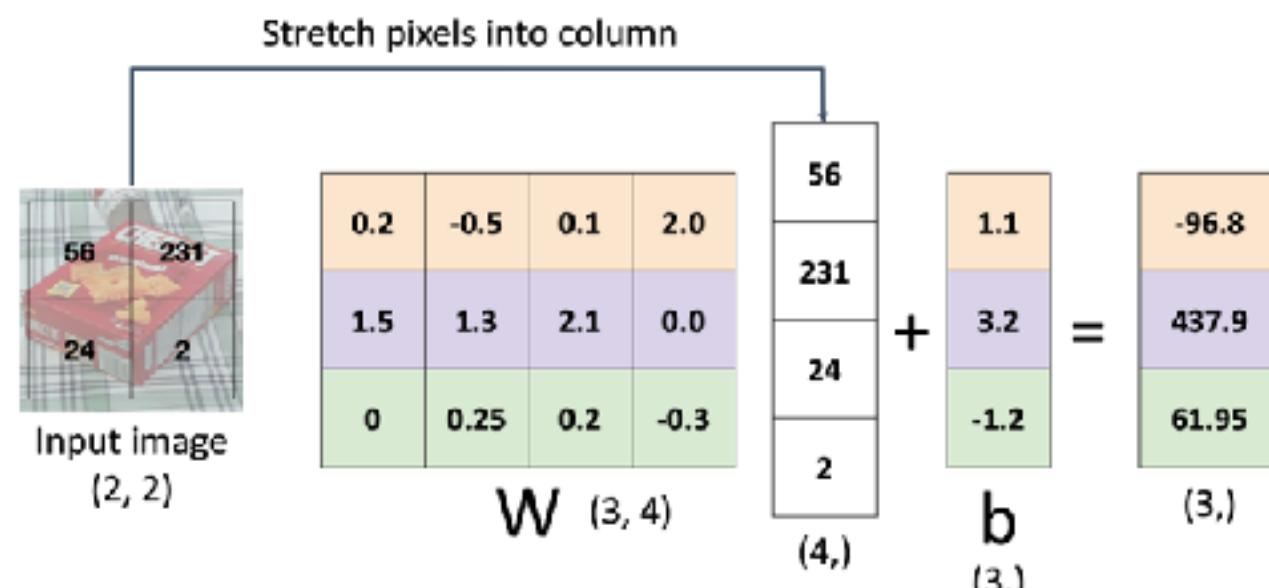
X	Y	F(x,y)
0	0	0
0	1	1
1	0	1
1	1	0



Linear Classifier—Three Viewpoints

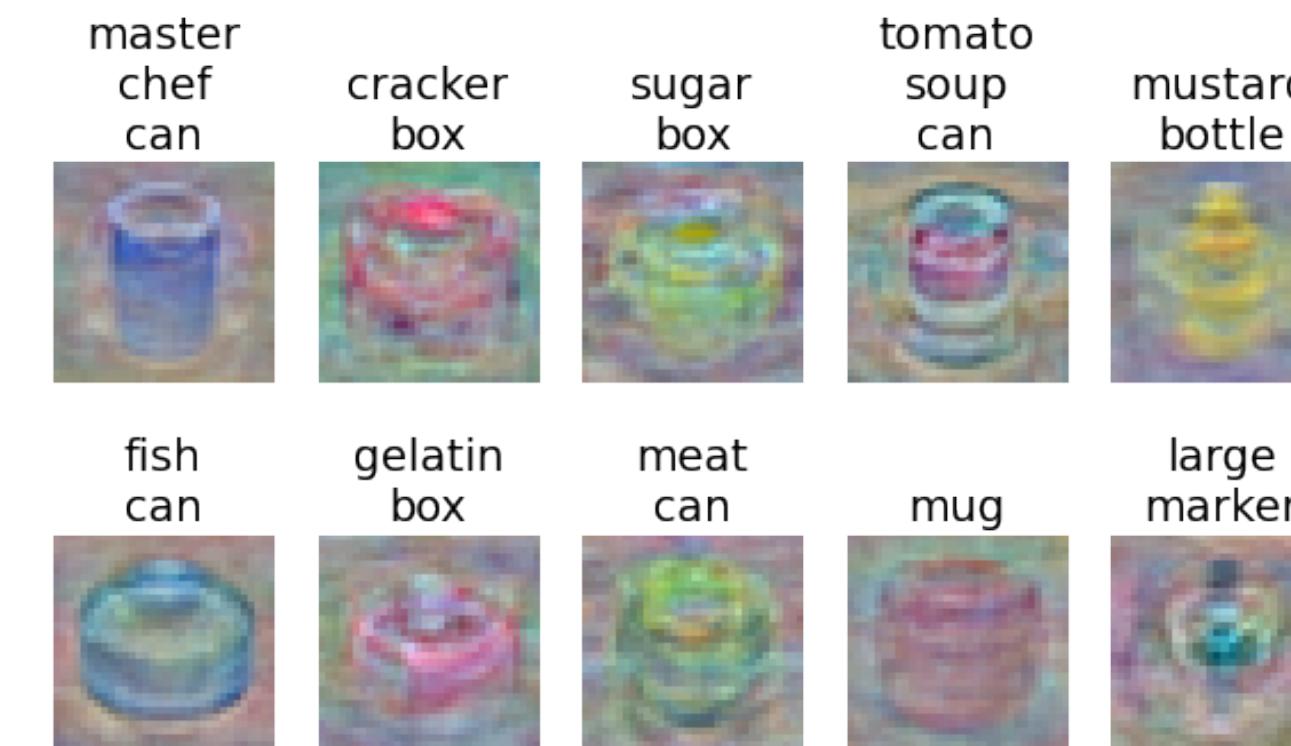
Algebraic Viewpoint

$$f(x, W) = Wx$$



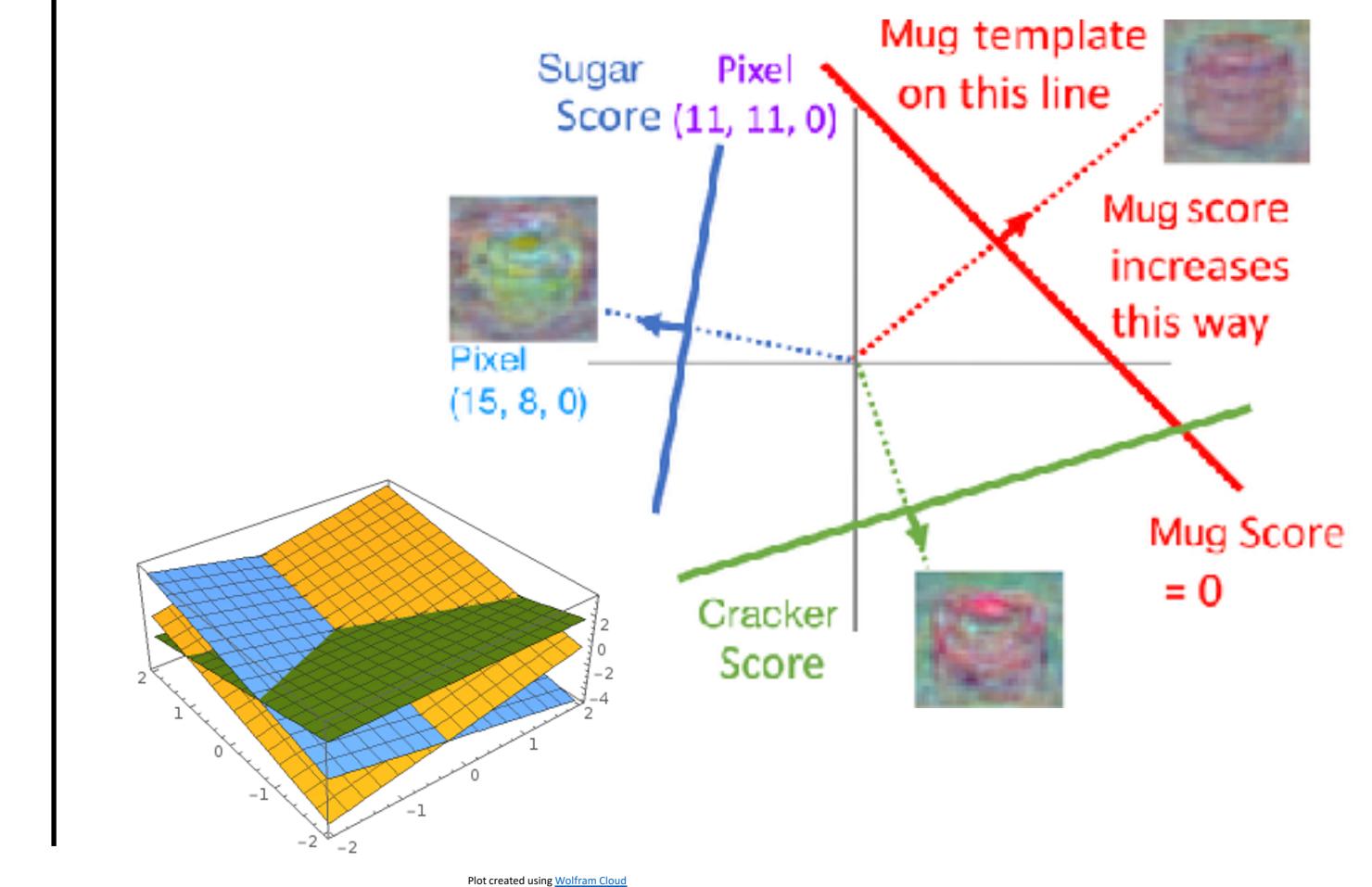
Visual Viewpoint

One template per class



Geometric Viewpoint

Hyperplanes cutting up space



So far—Defined a Score Function



$$f(x, W) = Wx + b$$

master chef can	-3.45	-0.51	3.42
mug	-8.87	6.04	4.64
tomato soup can	0.09	5.31	2.65
cracker box	2.9	-4.22	5.1
mustard bottle	4.48	-4.19	2.64
tuna fish can	8.02	3.58	5.55
sugar box	3.78	4.49	-4.34
gelatin box	1.06	-4.37	-1.5
potted meat can	-0.36	-2.09	-4.79
large marker	-0.72	-2.93	6.14

Given a W , we can compute class scores for an image, x .

But how can we actually choose a good W ?

So far—Choosing a Good W



master chef can	-3.45	-0.51	3.42
mug	-8.87	6.04	4.64
tomato soup can	0.09	5.31	2.65
cracker box	2.9	-4.22	5.1
mustard bottle	4.48	-4.19	2.64
tuna fish can	8.02	3.58	5.55
sugar box	3.78	4.49	-4.34
gelatin box	1.06	-4.37	-1.5
potted meat can	-0.36	-2.09	-4.79
large marker	-0.72	-2.93	6.14

$$f(x, W) = Wx + b$$

TODO:

1. Use a **loss function** to quantify how good a value of W is
2. Find a W that minimizes the loss function (**optimization**)



Loss Function

A **loss function** measures how good our current classifier is

Low loss = good classifier

High loss = bad classifier

Also called: **objective function**,
cost function





Loss Function

A **loss function** measures how good our current classifier is

Low loss = good classifier

High loss = bad classifier

Also called: **objective function**,
cost function

Negative loss function
sometimes called **reward function**, **profit function**, **utility function**, **fitness function**, etc.



Loss Function

A **loss function** measures how good our current classifier is

Low loss = good classifier

High loss = bad classifier

Also called: **objective function**,
cost function

Negative loss function
sometimes called **reward function**, **profit function**, **utility function**, **fitness function**, etc.

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^N$$

where x_i is an image and

y_i is a (discrete) label

Loss Function

A **loss function** measures how good our current classifier is

Low loss = good classifier

High loss = bad classifier

Also called: **objective function**,
cost function

Negative loss function
sometimes called **reward function**, **profit function**, **utility function**, **fitness function**, etc.

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^N$$

where x_i is an image and

y_i is a (discrete) label

Loss for a single example is

$$L_i(f(x_i, W), y_i)$$

Loss Function

A **loss function** measures how good our current classifier is

Low loss = good classifier

High loss = bad classifier

Also called: **objective function**,
cost function

Negative loss function
sometimes called **reward function**, **profit function**, **utility function**, **fitness function**, etc.

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^N$$

where x_i is an image and

y_i is a (discrete) label

Loss for a single example is

$$L_i(f(x_i, W), y_i)$$

Loss for the dataset is average of per-example losses:

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

Cross-Entropy Loss

Multinomial Logistic Regression



Want to interpret raw classifier scores as **probabilities**

cracker **3.2**

mug **5.1**

sugar **-1.7**

Cross-Entropy Loss

Multinomial Logistic Regression



Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \quad P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

cracker 3.2

mug 5.1

sugar -1.7

Cross-Entropy Loss

Multinomial Logistic Regression



Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \quad P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

cracker	3.2
mug	5.1
sugar	-1.7

Unnormalized log-probabilities (logits)



Cross-Entropy Loss

Multinomial Logistic Regression



cracker

3.2

mug

5.1

sugar

-1.7

Unnormalized log-
probabilities (logits)

$\exp(\cdot)$
→

24.5

164.0

0.18

Unnormalized
probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \quad P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities
must be ≥ 0

Cross-Entropy Loss

Multinomial Logistic Regression



cracker
mug
sugar

3.2
5.1
-1.7

Unnormalized log-probabilities (logits)

$\exp(\cdot)$

24.5
164.0
0.18

Unnormalized probabilities

normalize

0.13
0.87
0.00

Probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

Probabilities must be ≥ 0

$$P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Probabilities must sum to 1

Cross-Entropy Loss

Multinomial Logistic Regression



cracker
mug
sugar

3.2
5.1
-1.7

Unnormalized log-probabilities (logits)

$\exp(\cdot)$

24.5
164.0
0.18

Unnormalized probabilities

normalize

0.13
0.87
0.00

Probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

Probabilities must be ≥ 0

$$P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must sum to 1

$$L_i = -\log P(Y = y_i | X = x_i)$$

$$\begin{aligned} L_i &= -\log(0.13) \\ &= 2.04 \end{aligned}$$

Cross-Entropy Loss

Multinomial Logistic Regression



cracker

3.2

mug

5.1

sugar

-1.7

Unnormalized log-probabilities (logits)

$\exp(\cdot)$

24.5
164.0
0.18

Unnormalized probabilities

normalize

0.13
0.87
0.00

Probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

Probabilities
must be ≥ 0

$$P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities
must sum to 1

$$L_i = -\log P(Y = y_i | X = x_i)$$

$$\begin{aligned} L_i &= -\log(0.13) \\ &= 2.04 \end{aligned}$$

Maximum Likelihood Estimation

Choose weights to maximize the likelihood of the observed data
(see EECS 445 or EECS 545)

Cross-Entropy Loss

Multinomial Logistic Regression



cracker
mug
sugar

3.2
5.1
-1.7

Unnormalized log-probabilities (logits)

$\exp(\cdot)$

24.5
164.0
0.18

Unnormalized probabilities

normalize

0.13
0.87
0.00

Probabilities

compare

1.00
0.00
0.00

Correct probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

Probabilities must be ≥ 0

$$P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must sum to 1

Cross-Entropy Loss

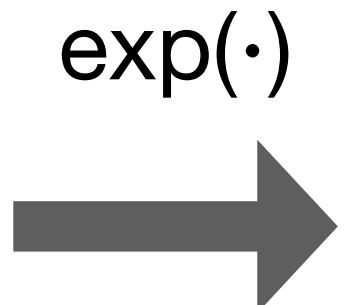
Multinomial Logistic Regression



cracker
mug
sugar

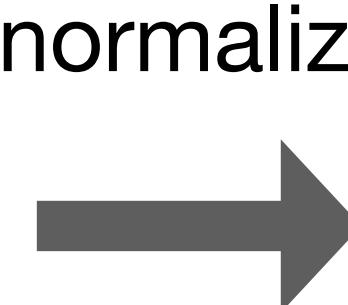
3.2
5.1
-1.7

Unnormalized log-probabilities (logits)



24.5
164.0
0.18

Unnormalized probabilities



0.13
0.87
0.00

Probabilities



1.00
0.00
0.00

Correct probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

Probabilities must be ≥ 0

$$P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must sum to 1

Kullback-Leibler divergence

$$D_{KL}(P || Q) =$$

$$\sum_y P(y) \log \frac{P(y)}{Q(y)}$$

Cross-Entropy Loss

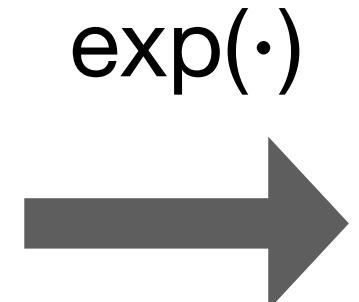
Multinomial Logistic Regression



cracker
mug
sugar

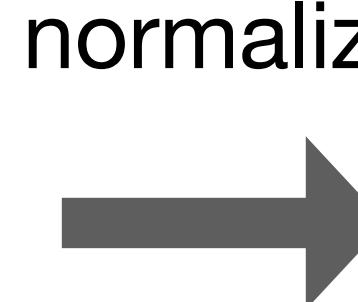
3.2
5.1
-1.7

Unnormalized log-probabilities (logits)



24.5
164.0
0.18

Unnormalized probabilities



0.13
0.87
0.00

Probabilities



1.00
0.00
0.00

Correct probabilities

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

Probabilities must be ≥ 0

Probabilities must sum to 1

$$P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Cross Entropy

$$H(P, Q) = H(P) + D_{KL}(P || Q)$$

Cross-Entropy Loss

Multinomial Logistic Regression



cracker	3.2
mug	5.1
sugar	-1.7

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \quad P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

$$L_i = -\log P(Y = y_i | X = x_i)$$

Putting it all together

$$L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$$

Cross-Entropy Loss

Multinomial Logistic Regression



cracker	3.2
mug	5.1
sugar	-1.7

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \quad P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

$$L_i = -\log P(Y = y_i | X = x_i)$$

Putting it all together

$$L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$$

Q: What is the min /
max possible loss L_i ?

Cross-Entropy Loss

Multinomial Logistic Regression



cracker	3.2
mug	5.1
sugar	-1.7

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \quad P(Y = k | X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

$$L_i = -\log P(Y = y_i | X = x_i)$$

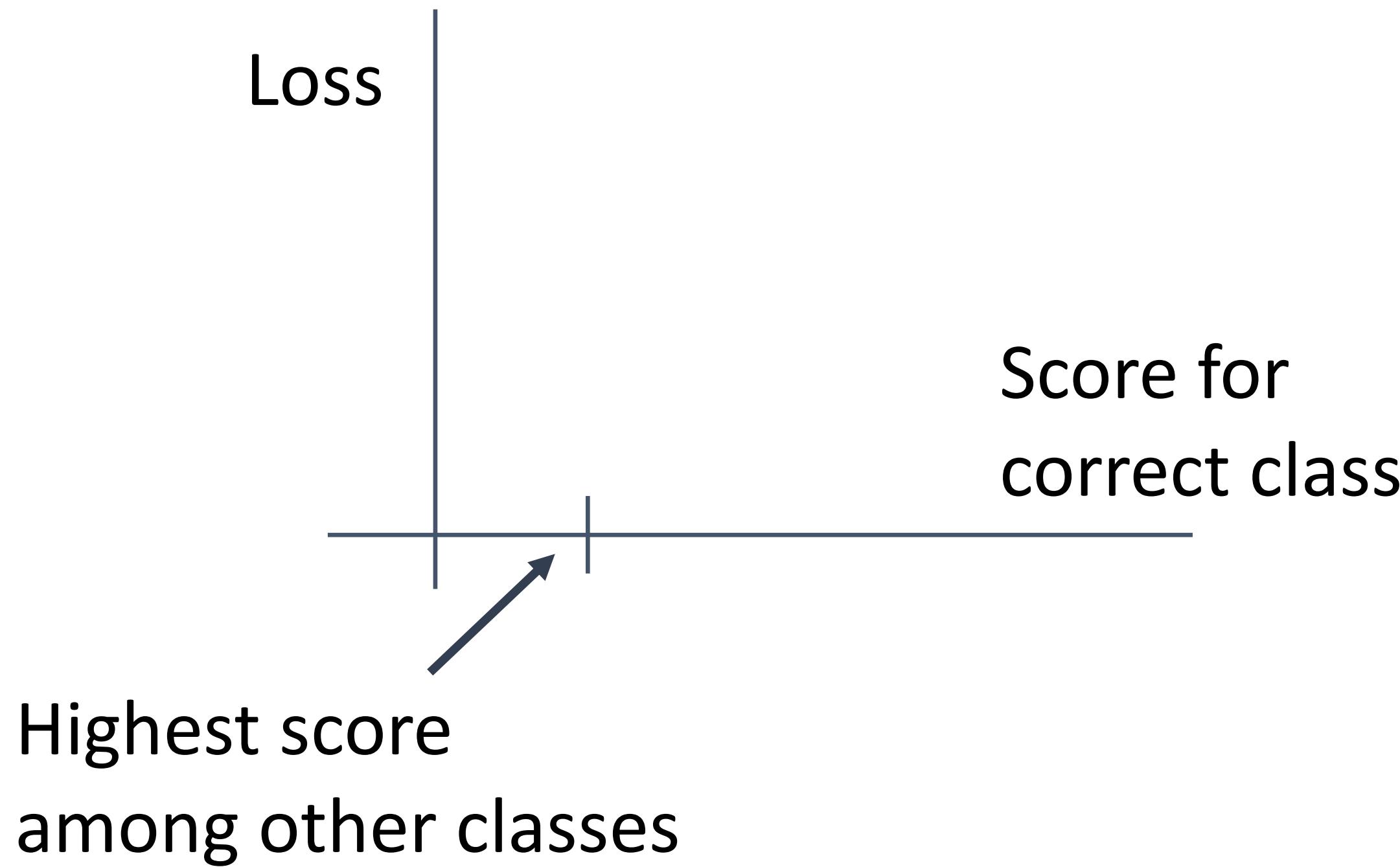
Putting it all together

$$L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$$

Q: If all scores are small random values, what is the loss?

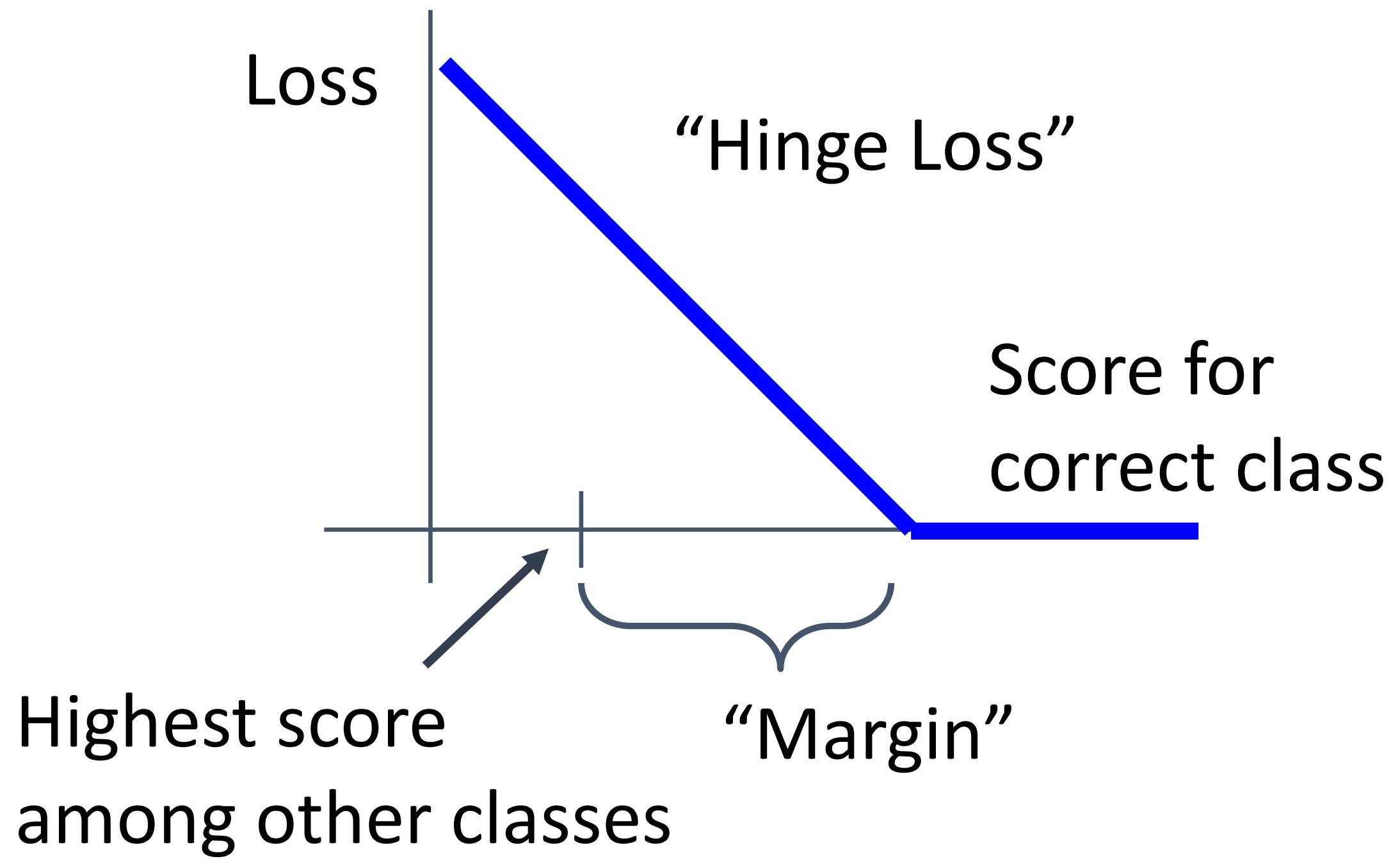
Multiclass SVM Loss

"The score of the correct class should be higher than all the other scores"



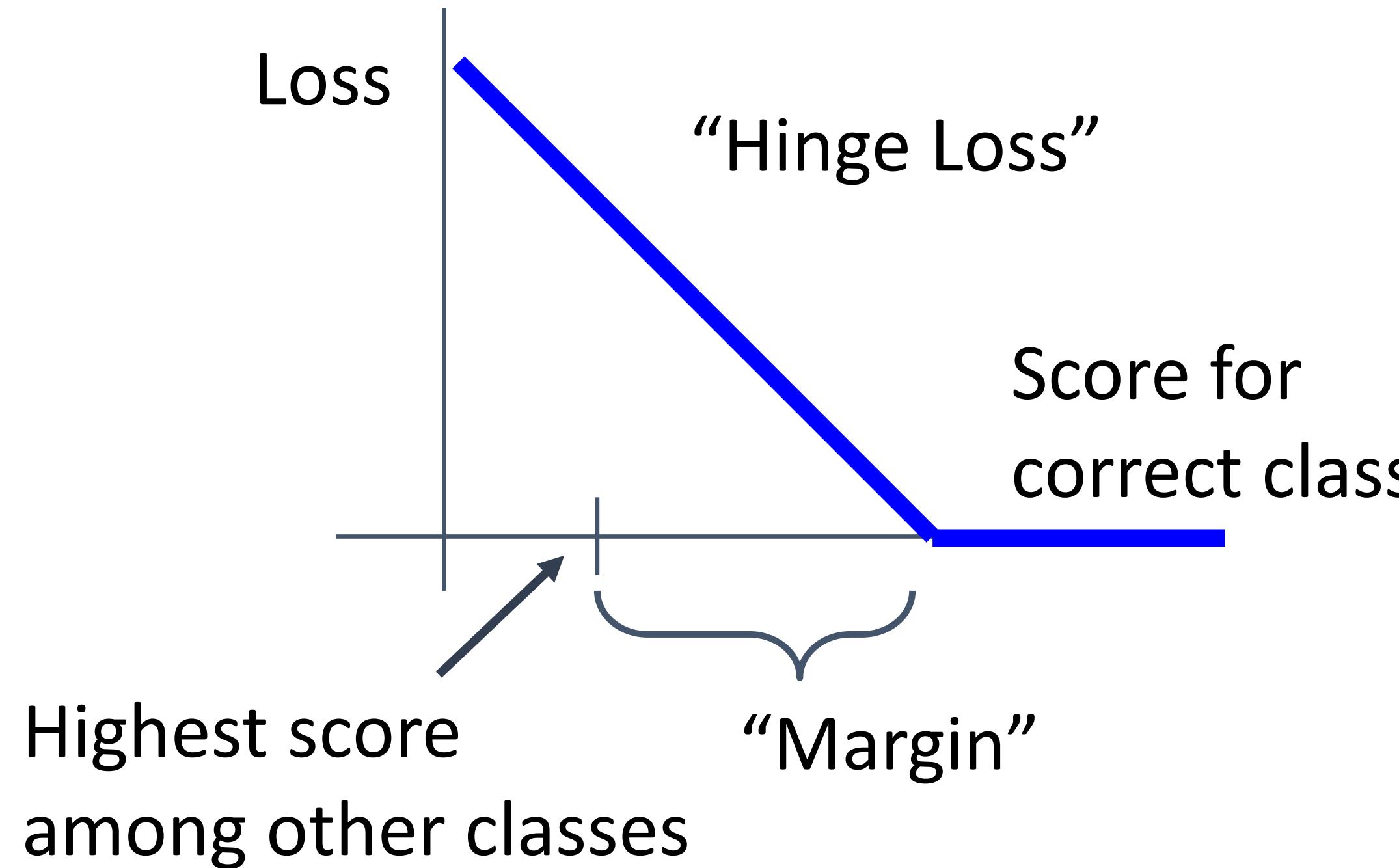
Multiclass SVM Loss

"The score of the correct class should be higher than all the other scores"



Multiclass SVM Loss

"The score of the correct class should be higher than all the other scores"



Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9		

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$\begin{aligned}
 L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\
 &= \max(0, 5.1 - 3.2 + 1) \\
 &\quad + \max(0, -1.7 - 3.2 + 1) \\
 &= \max(0, 2.9) + \max(0, -3.9) \\
 &= 2.9 + 0 \\
 &= 2.9
 \end{aligned}$$

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$\begin{aligned}
 L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\
 &= \max(0, 1.3 - 4.9 + 1) \\
 &\quad + \max(0, 2.0 - 4.9 + 1) \\
 &= \max(0, -2.6) + \max(0, -1.9) \\
 &= 0 + 0 \\
 &= 0
 \end{aligned}$$

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$\begin{aligned}
 L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\
 &= \max(0, 2.2 - (-3.1) + 1) \\
 &\quad + \max(0, 2.5 - (-3.1) + 1) \\
 &= \max(0, 6.3) + \max(0, 6.6) \\
 &= 6.3 + 6.6 \\
 &= 12.9
 \end{aligned}$$

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Loss over the dataset is:

$$\begin{aligned} L &= (2.9 + 0.0 + 12.9) / 3 \\ &= 5.27 \end{aligned}$$

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q: What happens to the loss if the scores for the car image change a bit?

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q2: What are the min
and max possible loss?

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q3: If all the scores were random, what loss would we expect?

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q4: What would happen if
the sum were over all
classes? (including $i = y_i$)

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q5: What if the loss used
a mean instead of a sum?

Multiclass SVM Loss



cracker	3.2	1.3	2.2
mug	5.1	4.9	2.5
sugar	-1.7	2.0	-3.1
Loss	2.9	0	12.9

Given an example (x_i, y_i)
 $(x_i$ is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q6: What if we used
this loss instead?

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)^2$$

Cross-Entropy vs SVM Loss

$$L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

and $y_i = 0$

Q: What is cross-entropy loss?
What is SVM loss?

Cross-Entropy vs SVM Loss

$$L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

and $y_i = 0$

Q: What happens to each loss if I slightly change the scores of the last datapoint?

Cross-Entropy vs SVM Loss

$$L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

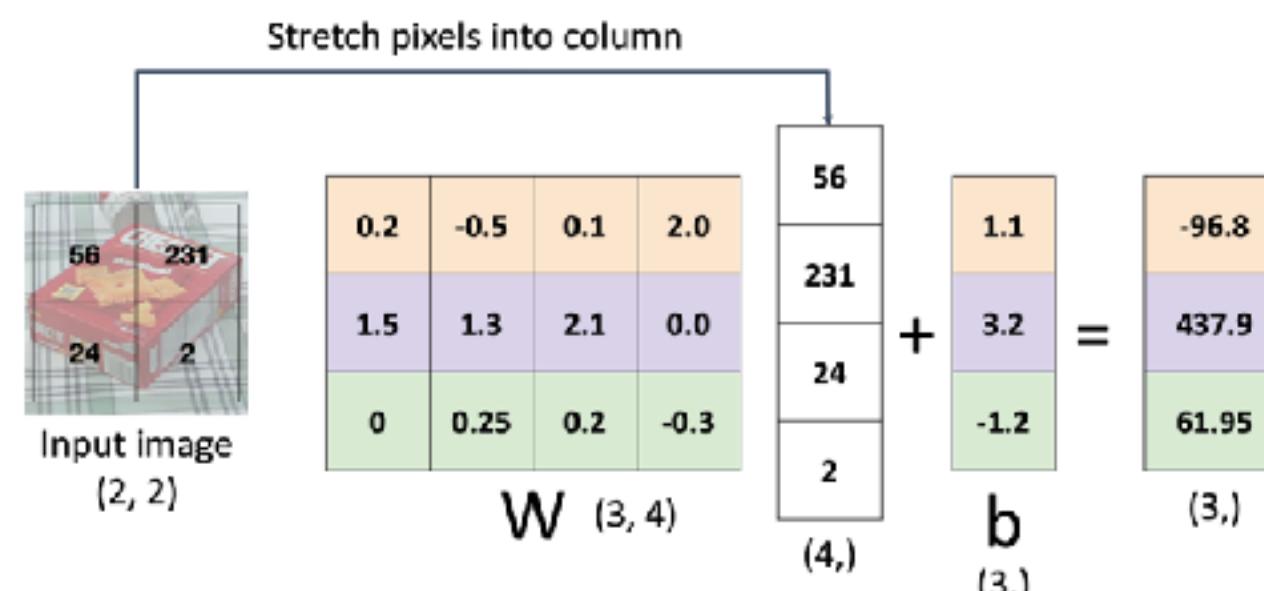
and $y_i = 0$

Q: What happens to each loss if I double the score of the correct class from 10 to 20?

Recap—Three Ways to Interpret Linear Classifiers

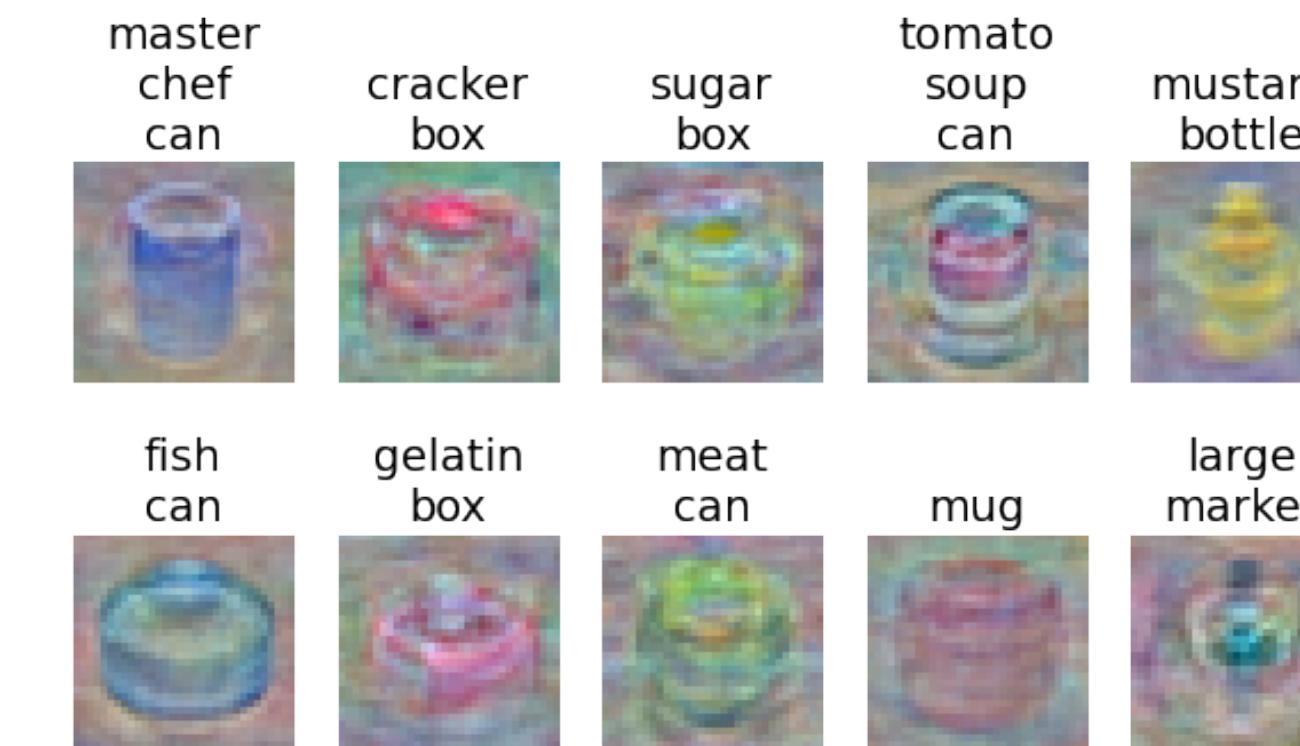
Algebraic Viewpoint

$$f(x, W) = Wx$$



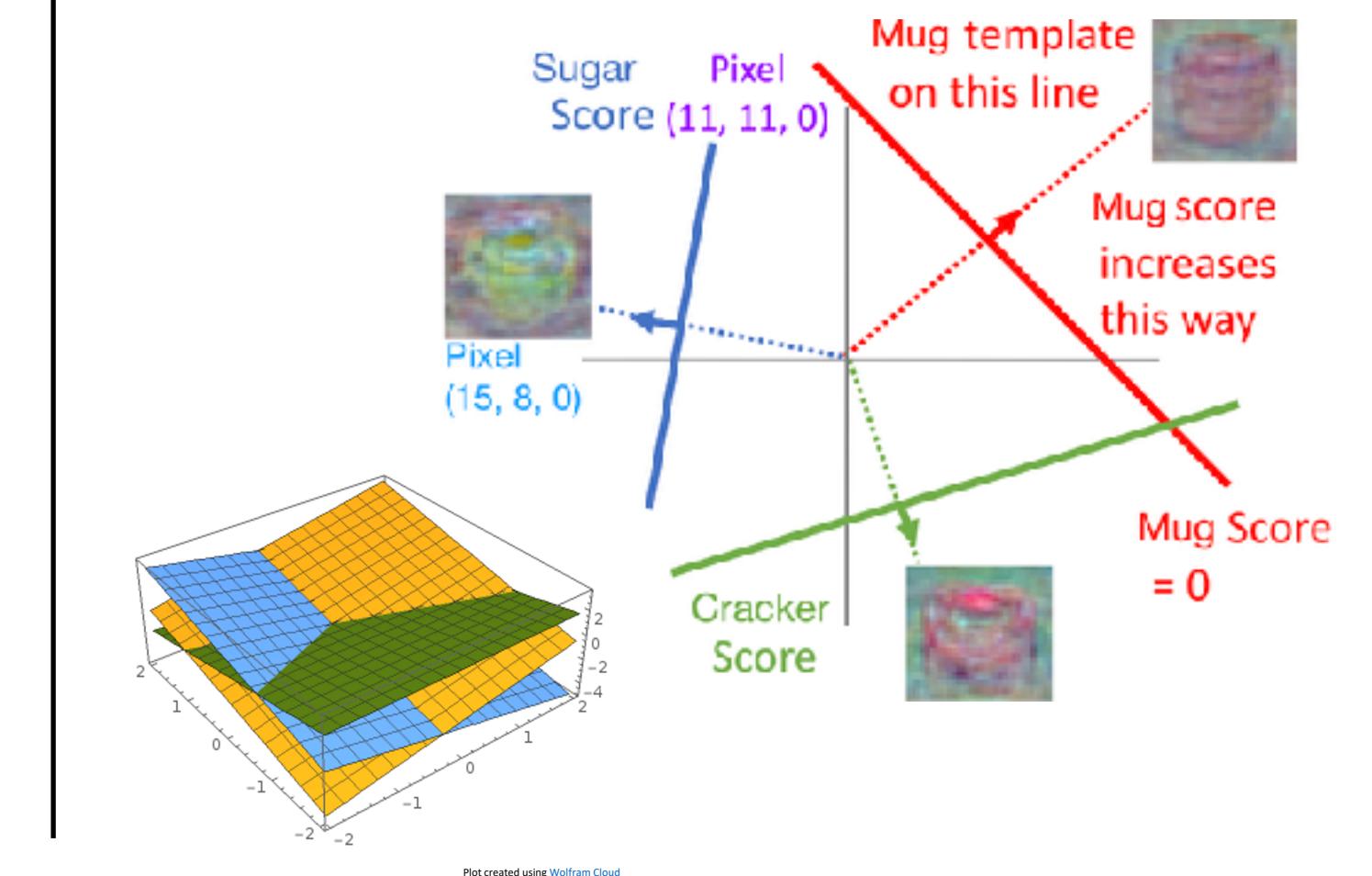
Visual Viewpoint

One template per class



Geometric Viewpoint

Hyperplanes cutting up space



Recap—Loss Functions Quantify Preferences

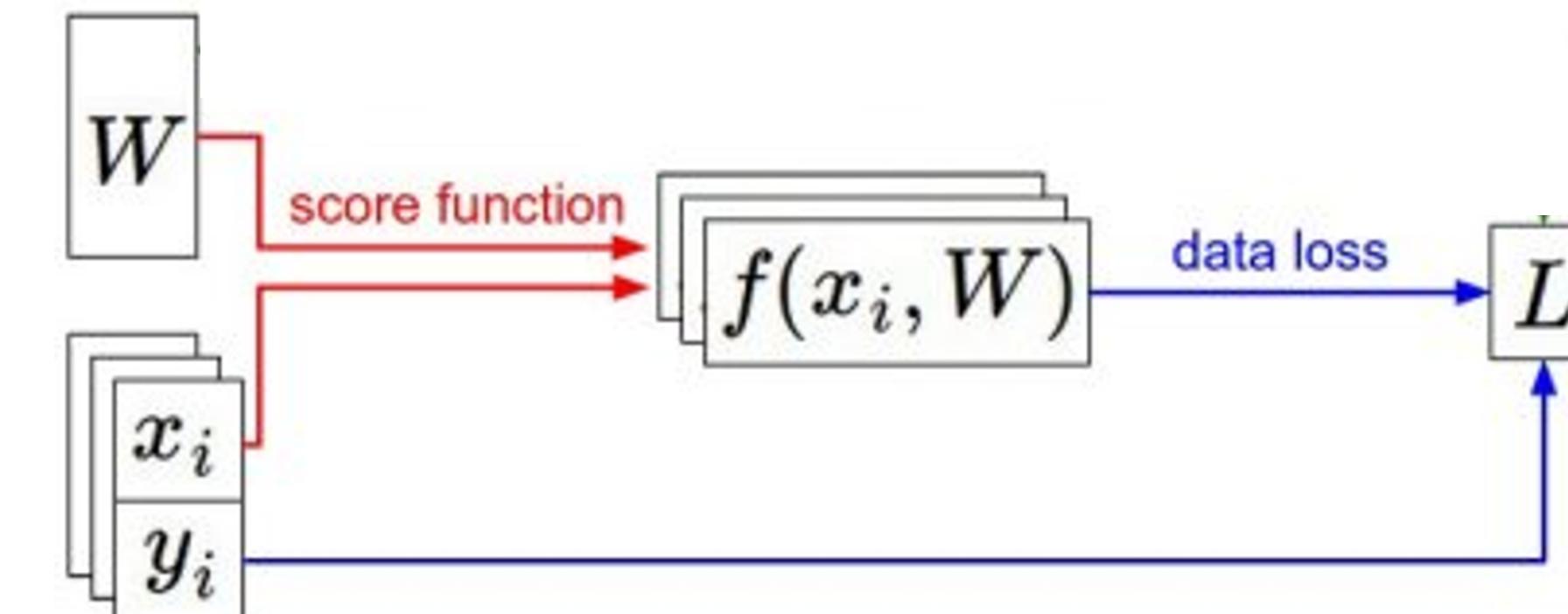
- We have some dataset of (x, y)
- We have a **score function**:
- We have a **loss function**:

$$s = f(x; W, b) = Wx + b$$

Linear classifier

Softmax: $L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$

SVM: $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$



Recap—Loss Functions Quantify Preferences

- We have some dataset of (x, y)
- We have a **score function**:
- We have a **loss function**:

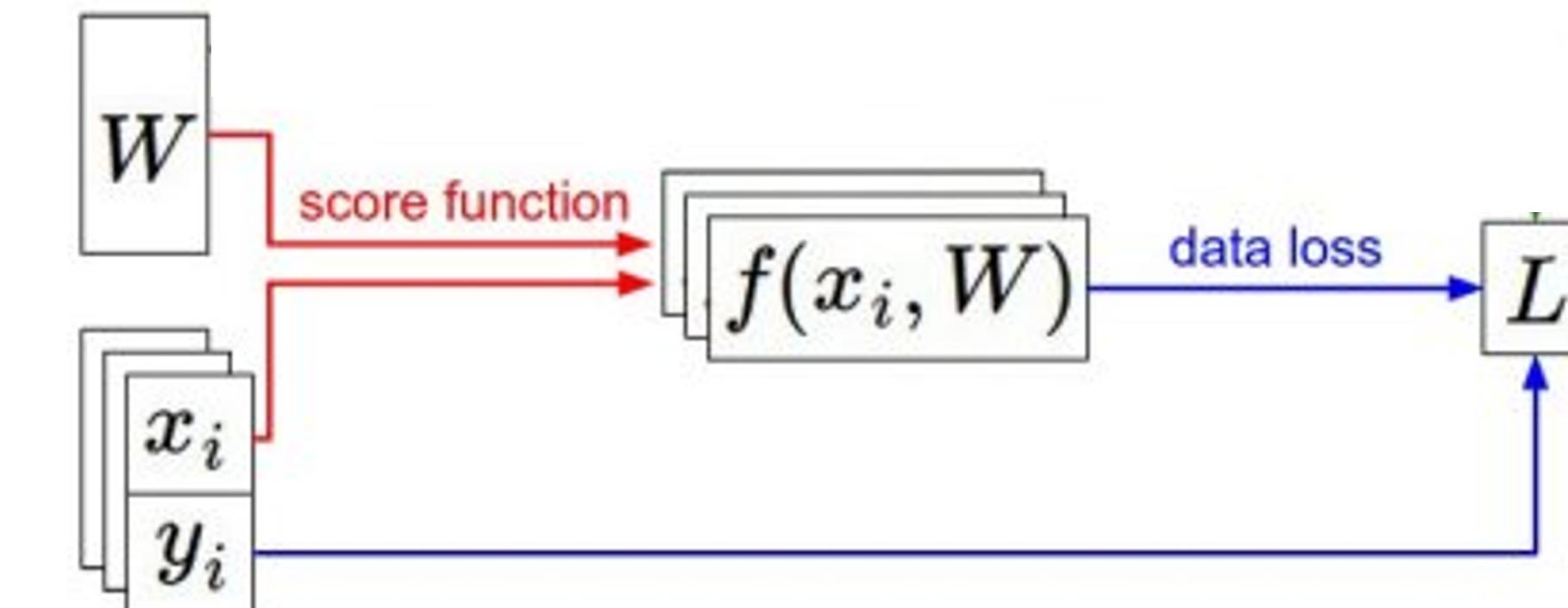
Q: How do we find the best W, b ?

$$s = f(x; W, b) = Wx + b$$

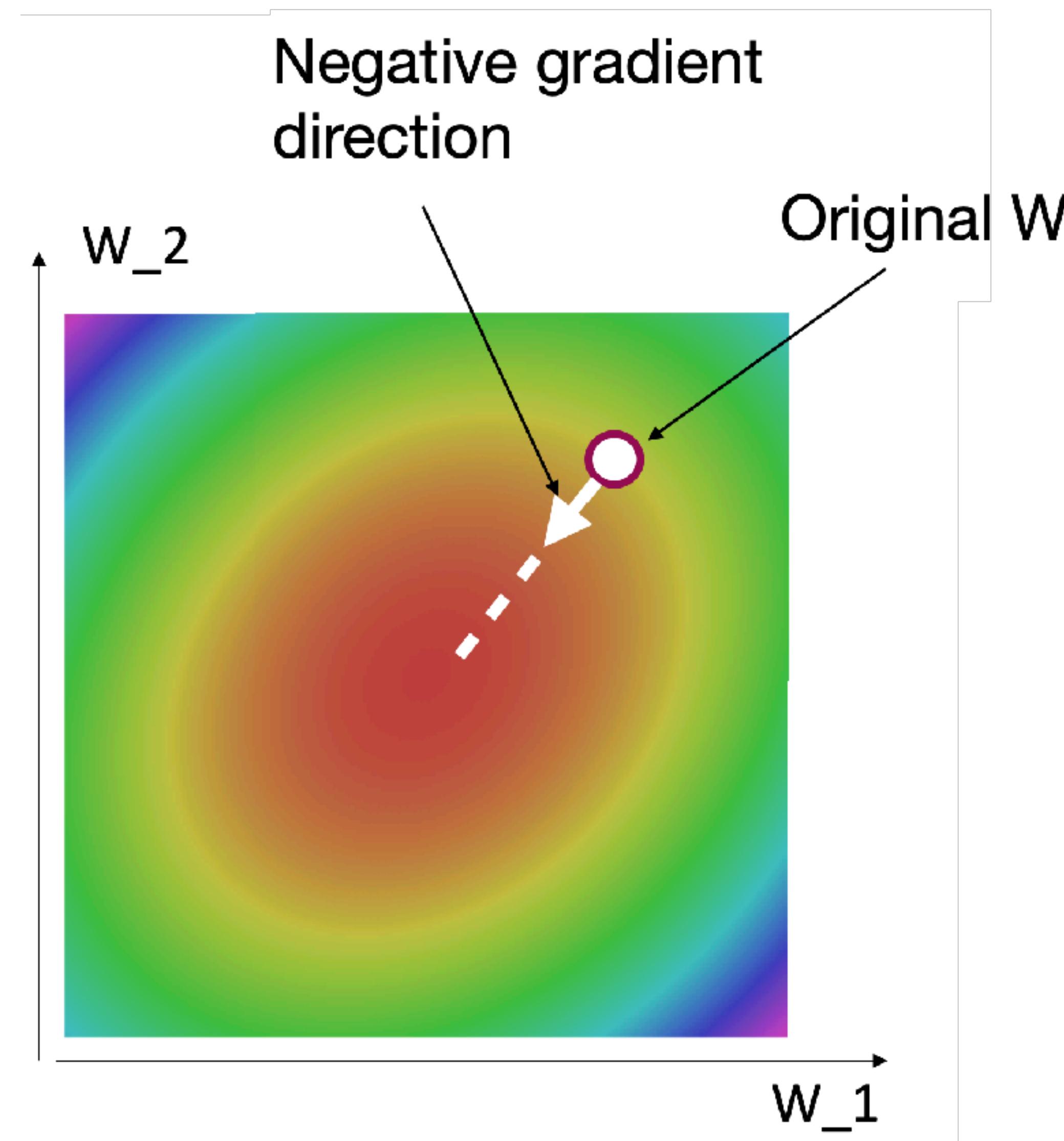
Linear classifier

Softmax: $L_i = -\log \left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)} \right)$

SVM: $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$



Next time: Regularization + Optimization



DR



DeepRob

Lecture 3
Linear Classifiers
University of Michigan and University of Minnesota

